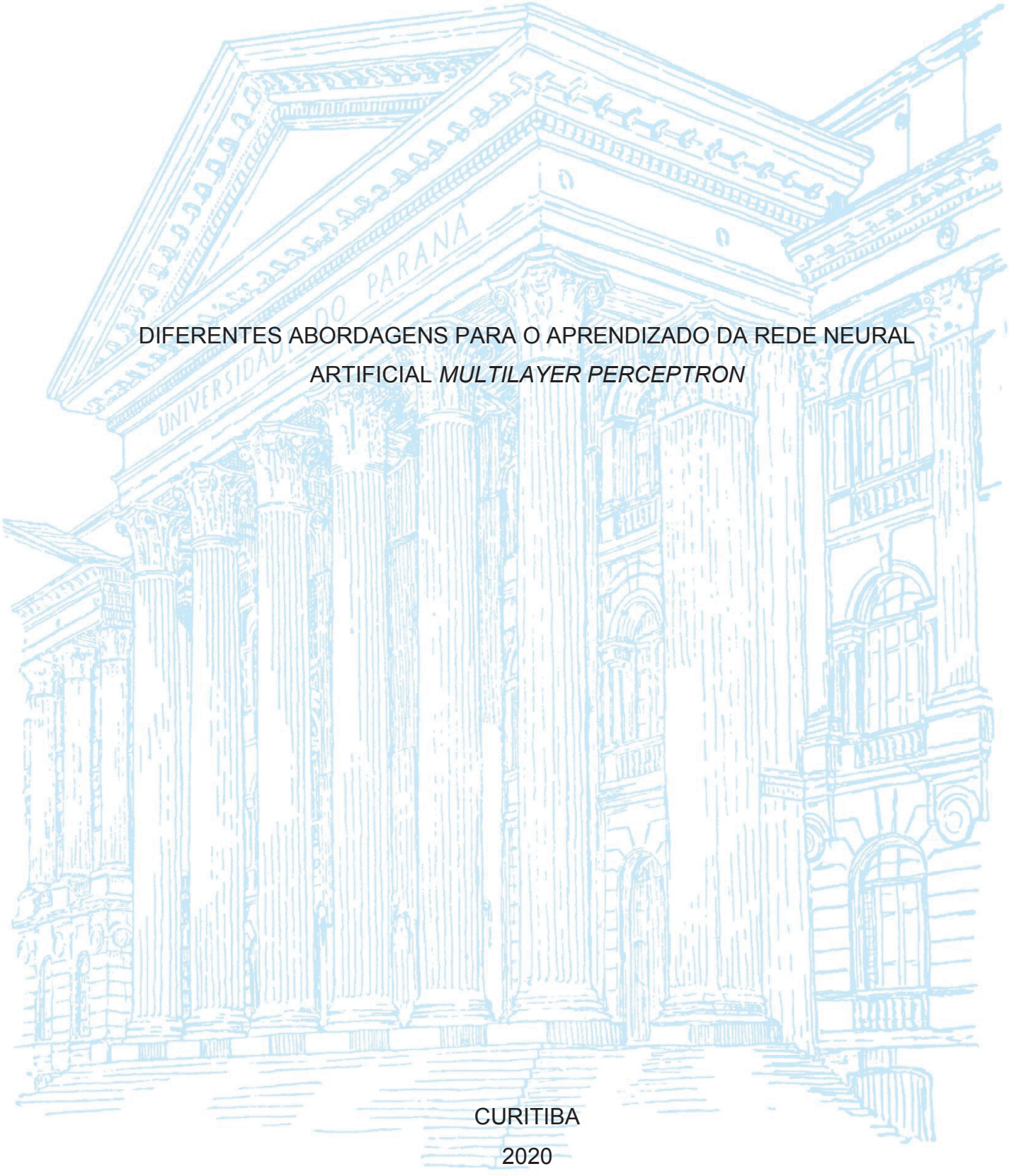
UNIVERSIDADE FEDERAL DO PARANÁ

SUELLEN TEIXEIRA ZAVADZKI DE PAULI



DIFERENTES ABORDAGENS PARA O APRENDIZADO DA REDE NEURAL ARTIFICIAL *MULTILAYER PERCEPTRON*

CURITIBA

2020

SUELLEN TEIXEIRA ZAVADZKI DE PAULI

DIFERENTES ABORDAGENS PARA O APRENDIZADO DA REDE NEURAL ARTIFICIAL *MULTILAYER PERCEPTRON*

Dissertação apresentada ao curso de Pós-Graduação em Engenharia de Produção, Setor de Tecnologia, Universidade Federal do Paraná, como requisito parcial à obtenção do título de Mestre em Engenharia de Produção.

Orientadora: Profª. Drª.Mariana Kleina

Coorientador: Prof. Dr. Wagner Hugo Bonat

CURITIBA

2020

CATALOGAÇÃO NA FONTE – SIBI/UFPR

P327d Pauli, Suellen Teixeira Zavadzki de

Diferentes abordagens para o aprendizado da rede neural artificial *multilayer perceptron* [recurso eletrônico/ Suellen Teixeira Zavadzki de Pauli. Curitiba, 2020.

Dissertação (Mestrado) - Pós-Graduação em Engenharia de Produção, Setor de Tecnologia, Universidade Federal do Paraná, como requisito parcial à obtenção do título de Mestre em Engenharia de Produção.
Orientadora: Profa. Dra.Mariana Kleina
Coorientador: Prof. Dr. Wagner Hugo Bonat

1. Inteligência computacional. 2. Redes neurais. I. Kleina, Mariana. II. Bonat, Wagner Hugo. I. Título.

CDD 621.399

Bibliotecária: Vilma Machado CRB9/1563

MINISTÉRIO DA EDUCAÇÃO
SETOR DE TECNOLOGIA
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO ENGENHARIA DE PRODUÇÃO - 40001016070P1

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em ENGENHARIA DE PRODUÇÃO da Universidade Federal do Paraná foram convocados para realizar a arguição da dissertação de Mestrado de **SUELLEN TEIXEIRA ZAVADZKI DE PAULI** intitulada: **DIFERENTES ABORDAGENS PARA O APRENDIZADO DA REDE NEURAL ARTIFICIAL *MULTILAYER PERCEPTRON***, sob orientação da Profa. Dra. MARIANA KLEINA, que após terem inquirido a aluna e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de mestre está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

CURITIBA, 27 de Fevereiro de 2020.

Mariana Kleina

MARIANA KLEINA

Presidente da Banca Examinadora (UNIVERSIDADE FEDERAL DO PARANÁ)

PAULO HENRIQUE SIQUEIRA

Avaliador Externo (UNIVERSIDADE FEDERAL DO PARANÁ)

ARINEI CARLOS LINDBECK DA SILVA

Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

Wagner Hugo Bonat

WAGNER HUGO BONAT

Coorientador - Avaliador Externo (UNIVERSIDADE FEDERAL DO PARANÁ)

AGRADECIMENTOS

Agradeço primeiramente a Deus, que tem me iluminado na busca pelo conhecimento, me dado força e abençoado com a oportunidade de ensinar, além de me proporcionar tudo o que vem na sequência deste agradecimento.

Agradeço aos meus pais, Jefferson Zavadzki e Sueli Flora Teixeira Zavadzki, por me ensinarem que devo conquistar o que desejo por meio do esforço, também agradeço pela educação que me deram e por serem meu suporte e referências de vida.

Agradeço às minhas irmãs, Fernanda Teixeira Zavadzki e Camila Teixeira Zavadzki, por sempre acreditaram em mim e me ouvirem em todos os momentos.

Agradeço ao meu marido, Peterson Roberto de Pauli, que me apoiou neste desafio do mestrado e sempre acreditou no meu potencial, também pela compreenção nas inúmeras horas de estudo e algumas vezes de cansaço e preocupação, além das explosões de ansiedade.

Agradeço aos professores do Programa de Pós Graduação em Engenharia de Produção (PPGEP) e do Departamento de Estatística pelo conhecimento transmitido durante o mestrado e a graduação, o qual tornou possível tratar de um tema que abrangesse as duas áreas.

Agradeço em especial aos professores, Mariana Kleina e Wagner Hugo Bonat, que sempre estiveram disponíveis para dar todo o suporte necessário, mesmo com as demais atividades inerentes da vida acadêmica e, principalmente, por serem profissionais nos quais eu posso me inspirar. Agradeço pela paciência nos ensinamentos.

Agradeço aos colegas do mestrado em Engenharia de Produção e do Laboratório de Estatística e Geoinformação (LEG) pelo conhecimento e estudo compartilhado.

Agradeço à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo incentivo financeiro, o que tornou possível a dedicação à este trabalho.

RESUMO

A área de *Machine Learning* tem ganhado destaque nos últimos tempos e as redes neurais artificiais estão entre as técnicas mais populares neste campo. Tais técnicas possuem a capacidade de aprendizado que ocorre no processo iterativo dos ajustes dos parâmetros. No presente trabalho, foram avaliadas diferentes abordagens para o aprendizado da rede neural *Multilayer Perceptron* (MLP), cuja arquitetura constitui-se das camadas de entrada, ocultas e de saída. As funções de ativação utilizadas para este estudo foram a sigmóide logística na camada oculta e a identidade para a de saída. Buscou-se compreender o aprendizado da rede inicialmente com dados simulados de uma estrutura de MLP, em que três algoritmos convencionais obtidos no pacote *neuralnet* do *software* R foram aplicados para a estimação dos parâmetros. Também foi feita a estimação via inferência Bayesiana e, ao final, uma nova proposta de aprendizado foi aplicada, a qual utiliza do *ranking* do *Score Information Criteria* (SIC) como ideia principal, para esta abordagem o algoritmo *simulated annealing* (SA) juntamente com a regressão linear são utilizados para a etapa de otimização. As mesmas técnicas também foram utilizadas para previsão em dois estudos de caso, sendo eles a previsão do preço de petróleo WTI e a previsão de exportação de produtos alimentícios, cuja unidade dos dados está em milhões de US$. A técnica proposta mostrou-se eficiente no modelo de séries temporais do petróleo, com acertividade compatível com a de técnicas tradicionais. O principal ganho está na simplicidade do modelo, com uma redução considerável no número de parâmetros.

Palavras-chave: Inteligência Computacional. Redes Neurais Artificiais. Algoritmos de aprendizado. *Score Information Criteria*.

ABSTRACT

The Machine Learning area has been gaining more attention and artificial neural networks are among the most well known techniques in this research area. Such techniques have the learning capacity that occurs in the iterative process of parameter adjustments. This paper evaluated different approaches for learning the Multilayer Perceptron (MLP) neural network, whose architecture consists of the input, hidden and output layers. The activation functions used for this study were the logistical sigmoid function in the hidden layer and the identity functions in the output layer. We sought to understand the network learning initially with simulated data from an MLP structure. Three conventional algorithms obtained in the neuralnet package of the R software were applied to estimate the parameters. Bayesian inference for parameters estimation was also used. Ultimately, a new learning proposal was applied, which uses the Score Information Criteria (SIC) ranking as the core concept. For this approach the Simulated Annealing (SA) algorithm and the linear regression were used for the optimization. The same techniques were also used to predict two case studies such as the WTI oil price prediction and the food product export prediction. The proposed technique proved to be efficient in the oil time series model. The main gain in this work is the simplicity of the model, with a reduction in the number of parameters.

Keywords: Computational Intelligence. Artificial neural networks. Learning algorithms. Score Information Criteria.

LISTA DE FIGURAS

FIGURA 1 - FILTROS DA PESQUISA 20
FIGURA 2 - MODELO DE NEURÔNIO 26
FIGURA 3 - EXEMPLOS GRÁFICOS DE FUNÇÃO DE ATIVAÇÃO 28
FIGURA 4 – REDE NEURAL *MULTILAYER PERCEPTRON* 30
FIGURA 5 –REDE NEURAL COM SAÍDA DE DISTRIBUIÇÃO NORMAL 35
FIGURA 6 - VEROSSIMILHANÇA DOS DADOS SIMULADOS 36
FIGURA 7 – ALGORITMO DE *BACKPROPAGATION* 38
FIGURA 8 – VALORES DE PARÂMETROS ESTIMADOS PELOS ALGORITMOS 40
FIGURA 9 - ERROS E ITERAÇÕES DOS ALGORITMOS NO TREINAMENTO 41
FIGURA 10 - VEROSSIMILHANÇA *RPROP+* 42
FIGURA 11 - VEROSSIMILHANÇA *RPROP -* 43
FIGURA 12 - VEROSSIMILHANÇA *SLR* 43
FIGURA 13 - DISTRIBUIÇÃO *A POSTERIORI* COM 1 NEURÔNIO 47
FIGURA 14 - CADEIAS DE MARKOV COM 1 NEURÔNIO 47
FIGURA 15 - DISTRIBUIÇÃO *A POSTERIORI* COM 2 NEURÔNIOS 48
FIGURA 16 - CADEIAS DE MARKOV COM 2 NEURÔNIOS 48
FIGURA 17 - DISTRIBUIÇÃO *A POSTERIORI* COM 3 NEURÔNIOS 49
FIGURA 18 - CADEIAS DE MARKOV COM 3 NEURÔNIOS 49
FIGURA 19 - DISTRIBUIÇÃO *A POSTERIORI* COM 4 NEURÔNIOS 50
FIGURA 20 - CADEIAS DE MARKOV COM 4 NEURÔNIOS 50
FIGURA 21 - DISTRIBUIÇÃO *A POSTERIORI* COM 5 NEURÔNIOS 51
FIGURA 22 - CADEIAS DE MARKOV COM 5 NEURÔNIOS 51
FIGURA 23 - CONFIGURAÇÃO INICIAL DO MODELO POPOSTO 56
FIGURA 24 – PSEUDOCÓDIGO DO ALGORITMO PROPOSTO 57
FIGURA 25 – ERROS DO MODELO SIC 59
FIGURA 26 - CLASSIFICAÇÃO DA PESQUISA 60
FIGURA 27 - ETAPAS DA PESQUISA 61
FIGURA 28 - SÉRIE HISTÓRICA DE PETRÓLEO WTI 62
FIGURA 29 - SÉRIE HISTÓRICA DE EXPORTAÇÃO DE ALIMENTOS 63
FIGURA 30 - CORRELAÇÃO PETRÓLEO 64
FIGURA 31 - CORRELAÇÃO EXPORTAÇÃO DE ALIMENTOS 65
FIGURA 32 - BOXPLOT PARA ERRO E ITERAÇÕES 67

FIGURA 33 - PREVISÃO ALGORITMOS RNA 67
FIGURA 34 – DISTRIBUIÇÃO *A POSTERIORI* COM 1 NEURÔNIO 68
FIGURA 35 - CADEIAS DE MARKOV COM 1 NEURÔNIO 68
FIGURA 36 - INTERVALO DE CREDIBILIDADE COM 1 NEURÔNIO 69
FIGURA 37 – DISTRIBUIÇÃO *A POSTERIORI* COM 2 NEURÔNIOS 69
FIGURA 38 - CADEIAS DE MARKOV COM 2 NEURÔNIOS.....70
FIGURA 39 - INTERVALO DE CREDIBILIDADE COM 2 NEURÔNIOS 70
FIGURA 40 - DISTRIBUIÇÃO *A POSTERIORI* COM 3 NEURÔNIOS 71
FIGURA 41 - CADEIAS DE MARKOV COM 3 NEURÔNIOS.....71
FIGURA 42 - INTERVALO DE CREDIBILIDADE COM 3 NEURÔNIOS 72
FIGURA 43 - DISTRIBUIÇÃO *A POSTERIORI* COM 4 NEURÔNIOS 72
FIGURA 44 - CADEIAS DE MARKOV COM 4 NEURÔNIOS.....73
FIGURA 45 - INTERVALO DE CREDIBILIDADE COM 4 NEURÔNIOS 73
FIGURA 46 - DISTRIBUIÇÃO *A POSTERIORI* COM 5 NEURÔNIOS 74
FIGURA 47 - CADEIAS DE MARKOV COM 5 NEURÔNIOS.....74
FIGURA 48 - INTERVALO DE CREDIBILIDADE COM 5 NEURÔNIOS 75
FIGURA 49 - ERROS DO MODELO 77
FIGURA 50 - PREVISÃO COM SIC 77
FIGURA 51 – ERROS E ITERAÇÕES DOS ALGORITMOS TRADICIONAIS 79
FIGURA 52 - PREVISÃO DOS ALGORITMOS TRADICIONAIS 79
FIGURA 53 - DISTRIBUIÇÃO *A POSTERIORI* COM 1 NEURÔNIO 80
FIGURA 54 - CADEIAS DE MARKOV COM 1 NEURÔNIO 80
FIGURA 55 - INTERVALO DE CREDIBILIDADE COM 1 NEURÔNIO 81
FIGURA 56 - DISTRIBUIÇÃO *A POSTERIORI* COM 2 NEURÔNIOS 81
FIGURA 57 - CADEIAS DE MARKOV COM 2 NEURÔNIOS.....82
FIGURA 58 - INTERVALO DE CREDIBILIDADE COM 2 NEURÔNIOS 82
FIGURA 59 - DISTRIBUIÇÃO *A POSTERIORI* COM 3 NEURÔNIOS 83
FIGURA 60 - CADEIAS DE MARKOV COM 3 NEURÔNIOS.....83
FIGURA 61 - INTERVALO DE CREDIBILIDADE COM 3 NEURÔNIOS 84
FIGURA 62 - DISTRIBUIÇÃO *A POSTERIORI* COM 4 NEURÔNIOS 84
FIGURA 63 - CADEIAS DE MARKOV COM 4 NEURÔNIOS.....85
FIGURA 64 - INTERVALO DE CREDIBILIDADE COM 4 NEURÔNIOS 85
FIGURA 65 - DISTRIBUIÇÃO *A POSTERIORI* COM 5 NEURÔNIOS 86
FIGURA 66 - CADEIAS DE MARKOV COM 5 NEURÔNIOS.....86

FIGURA 67 - INTERVALO DE CREDIBILIDADE COM 5 NEURÔNIOS 87
FIGURA 68 - ERROS DO MODELO PROPOSTO.....89
FIGURA 69 - PREVISÃO COM O SIC 89

LISTA DE QUADROS

QUADRO 1 - DETALHAMENTO DAS VARIÁVEIS NO *SOFTWARE* R.....63

LISTA DE TABELAS

TABELA 1 - EXEMPLOS DE FUNÇÃO DE ATIVAÇÃO.....27
TABELA 2 – PARÂMETROS ESTIMADOS QUE RESULTARAM EM UM MENOR ERRO.....42
TABELA 3 - CORRELAÇÃO DAS COVARIÁVEIS COM A VARIÁVEL RESPOSTA 58
TABELA 4 - *RANKING SIC* PRIMEIRO MODELO 58
TABELA 5 – MENORES ERROS DOS ALGORITMOS NO ESTUDO DE CASO 1 ..66
TABELA 6– CORRELAÇÃO COM A VARIÁVEL RESPOSTA.....75
TABELA 7– *RANKING* SIC POR NEURÔNIO 76
TABELA 8 – MENORES ERROS DOS ALGORITMOS NO ESTUDO DE CASO 2 ..78
TABELA 9– CORRELAÇÃO COM A VARIÁVEL RESPOSTA.....87
TABELA 10– *RANKING SIC* PRIMEIRO MODELO 88

LISTA DE ABREVIATURAS OU SIGLAS

EMV	- Estimador de Máxima Verossimilhança
GEE	- *Generalized Estimating Equations*
IPCA	- Índice Nacional de Preços ao Consumidor Amplo
JAGS	- *Just Another Gibbs Sampler*
MCMC	- Monte Carlo via Cadeia de Markov
MLP	- *Multilayer Perceptron*
MQ	- Mínimos Quadrados
RNA	- Redes Neurais Artificiais
SA	- *Simulated Annealing*
SIC	- *Score Information Criteria*
SQE	- Soma dos Quadrados dos Erros

LISTA DE SÍMBOLOS

Σ - somatório de números

Π - produtório de números

Δ - gradiente

$l(.)$ - log-verossimilhança

$f(.)$ – função densidade de probabilidade

$\frac{\partial .}{\partial .}$ - derivada

$\sim$ - segue uma distribuição de probabilidade

$C_{n,p}$ - combinação de n elementos p a p

$\propto$ - proporcional a

J - matriz jacobiana

M^{-1} - matriz inversa

M^{T} - matriz transposta

SUMÁRIO

1 INTRODUÇÃO 16
1.1 OBJETIVOS 17
1.1.1 Objetivo geral 17
1.1.2 Objetivos específicos 17
1.2 JUSTIFICATIVA 18
1.3 LIMITAÇÕES DO TRABALHO 18
1.4 ESTRUTURA DO TRABALHO 19
2 REVISÃO DA LITERATURA 20
2.1 PROTOCOLO DA PESQUISA 20
2.2 TRABALHOS CORRELATOS 21
2.3 REFERENCIAL TEÓRICO 25
2.3.1 REDES NEURAIS ARTIFICIAIS 25
2.3.1.1 Aprendizagem de uma Rede Neural Artificial 28
2.3.1.2 Modelo Multilayer Perceptron 29
2.3.2 SIMULAÇÃO DOS DADOS 31
2.3.3 ESTIMAÇÃO E INFERÊNCIA 32
2.3.4 INFERÊNCIA BAYESIANA 44
2.3.5 SIMULATED ANNEALING 52
2.3.6 SCORE INFORMATION CRITERIA 53
3 MATERIAL E MÉTODOS 60
3.1 CLASSIFICAÇÃO DA PESQUISA 60
3.2 ETAPAS DA PESQUISA 61
4 APRESENTAÇÃO DOS RESULTADOS 66
4.1 ESTUDO DE CASO 1: PREÇO DE PETRÓLEO WTI 66
4.2 ESTUDO DE CASO 2: EXPORTAÇÃO NO SETOR ALIMENTÍCIO 78
5 CONSIDERAÇÕES FINAIS 91
5.1 RECOMENDAÇÕES PARA TRABALHOS FUTUROS 93
REFERÊNCIAS 94
APÊNDICE 1 – DIAGRAMA DE DISPERSÃO RPROP + E RPROP- 99
APÊNDICE 2 – DIAGRAMA DE DISPERSÃO SLR E RPROP- 100
APÊNDICE 3 – DIAGRAMA DE DISPERSÃO *SLR* E *RPROP* + ... 101

1 INTRODUÇÃO

As Redes Neurais Artificiais (RNAs) são técnicas populares de *Machine Learning* que simulam o mecanismo de aprendizado em organismos biológicos. Uma RNA faz o cálculo por meio de uma função das entradas propagando os valores obtidos para os neurônios de saída e utiliza pesos como parâmetros intermediários. O aprendizado ocorre com a alteração dos pesos que conectam os neurônios a fim de reduzir os erros do modelo. Após o aprendizado da rede é possível generalizar o conhecimento obtido no treinamento para dados não conhecidos (AGGARWAL, 2018).

Uma arquitetura básica de rede neural, denominada *Perceptron,* constitui de variáveis de entrada diretamente relacionadas com a camada de saída. Já quando há camadas intermediárias, denominadas ocultas, tem-se a rede neural *Multilayer Perceptron*, a qual será abordada no presente trabalho.

Na rede neural *Perceptron*, que possui camada única, o processo de treinamento é relativamente direto pois a função de perda, ou seja, o erro, pode ser calculada como uma função direta dos pesos, o que permite facilmente calcular o gradiente. Já no caso da *Multilayer Perceptron*, a função perda é uma composição mais complexa. O treinamento desta rede ocorre nas fases de *forward* e *backward*. Na primeira são inseridas as entradas e estas passam por toda a estrutura de cálculos da rede com um determinado conjunto de pesos, ao final, o valor real é comparado ao obtido no treinamento. Já na segunda fase, o principal objetivo é aprender a rede por meio de um algoritmo de treinamento e com isso deseja-se reduzir a função de perda no decorrer das atualizações dos pesos (AGGARWAL, 2018).

Métodos de gradiente descendente são bastante utilizados para o aprendizado da rede. O algoritmo *backpropagation* (RUMELHART; MCCLELLEND,1986) é o mais popular desta categoria. Algoritmos adaptativos baseados no gradiente também foram propostos a fim de superar a dificuldade em escolher as melhores taxas de aprendizado para cada região no espaço de busca. Neste contexto, o algoritmo *Resilient backpropagation* (*Rprop*) (RIEDMILLER; BRAUN, 1993) ganhou destaque e motivou o desenvolvimento de diversas variantes, como *Grprop* (ANASTASIADIS *et al*., 2005).

Para o desenvolvimento deste trabalho, os algoritmos *backpropagation, Rprop* e *Grprop*, que estão disponíveis do pacote *neuralnet*, do *software* R, foram aplicados

em dois contextos. Inicialmente em dados simulados de uma RNA *Multilayer Perceptron* e posteriormente em dois conjuntos de dados, a fim de compreender melhor tais algoritmos de aprendizado. Nos mesmos dados simulados também foi aplicada inferência Bayesiana com o algoritmo de Monte Carlo via Cadeias de Markov, igualmente disponível no *software* R.

Por fim, tem-se uma proposta de algoritmo baseado no *Score Information Criteria* com uma abordagem no contexto de aprendizado das redes neurais *Multilayer Perceptron*. A técnica consiste na seleção de quais neurônios da camada de entrada permanecerão no modelo final, a otimização dos parâmetros foi feita com o algoritmo *simulated annealing* (SA) juntamente com a regressão linear.

1.1 OBJETIVOS

Nesta sessão serão apresentados os objetivos geral e específicos deste trabalho.

1.1.1 Objetivo geral

O objetivo geral deste trabalho é avaliar diferentes abordagens de aprendizado em uma rede neural *Multilayer Perceptron* aplicando as técnicas convencionais de aprendizado, a inferência Bayesiana e por fim um método baseado no *Score Information Criteria*.

1.1.2 Objetivos específicos

A fim de cumprir o objetivo geral, são propostos os seguintes objetivos específicos:

- Descrever as técnicas convencionais de estimação de parâmetros da rede neural *Multilayer Perceptron*, contidas no pacote *neuralnet*;
- Aplicar inferência Bayesiana para a estimação dos parâmetros e nesta abordagem construir um intervalo de credibilidade para a previsão;
- Propor um algoritmo para a estimação e seleção de parâmetros da rede neural com base na estatística de *Score Information Criteria*;

- Compreender as características das técnicas aplicadas com dados simulados de uma rede neural MLP;
- Avaliar as melhores técnicas em dois estudos de casos.

1.2 JUSTIFICATIVA

A previsão de eventos futuros é um fator importante para o planejamento de processos e para a tomada de decisão, como por exemplo, no monitoramento e ajuste de processos industriais, previsões de vendas de produtos ou demanda, controle de estoques, gerenciamento da cadeia de suprimento, gestão de risco financeiro em investimentos, entre outros. Um bom método de previsão pode se tornar uma poderosa ferramenta de auxílio na tomada de decisão (MONTGOMERY et al., 2008).

As Redes Neurais Artificiais fazem parte do grupo de técnicas de *Machine Learning*, as quais são consideradas promissoras no mundo atual, porém, em problemas reais, podem ser de difícil treinamento devido ao elevado número de parâmetros. O presente trabalho visa estudar a substituição do treinamento convencional da RNA *Multilayer Perceptron* para uma proposta de ajuste dos parâmetros da rede com redução no volume dos mesmos.

Com o objetivo de fundamentar o trabalho como competência de um Engenheiro de Produção, se verifica que de acordo com a Associação Brasileira de Engenharia de Produção (1998), cabe aos profissionais de Engenharia de Produção conteúdos profissionais relacionados à Pesquisa Operacional, nas subáreas Programação Matemática, Processos Estocásticos e Modelagem, Análise e Simulação. Para a aplicação optou-se por uma base de dados pública, as quais contemplam uma série histórica de preços de petróleo WTI e dados mensais, em milhões de US$, de exportação no setor alimentício no Brasil. As técnicas aplicadas têm a contribuição de auxiliar na tomada de decisão.

1.3 LIMITAÇÕES DO TRABALHO

A presente pesquisa visa agregar a flexibilidade das redes neurais à aplicações de um contexto de regressão, em que tem-se uma variável resposta e esta pode ser explicada por covariáveis. Sendo assim, a pesquisa fica restrita a este tipo

de contexto e a intenção não é de generalizar esta aplicação para qualquer tipo de problema.

1.4 ESTRUTURA DO TRABALHO

Com a finalidade de alcançar os objetivos previamente delineados na estruturação deste trabalho, tem-se a estrutura segmentada em cinco capítulos.

Neste primeiro capítulo foram apresentados os elementos relacionados ao planejamento desta pesquisa, incluindo a introdução a respeito do tema, seus objetivos, justificativa e limitações.

No capítulo 2, tem-se o referencial teórico, que constitui da descrição das técnicas utilizadas nas aplicações. Para melhor esplanar os conceitos, foram utilizados dados simulados ilustrando tais explicações.

A classificação da pesquisa está descrita no capítulo 3, no qual tem-se também as etapas consideradas no decorrer do trabalho com uma breve descrição das aplicações utilizadas, que constituem em dois estudos de caso.

No capítulo 4 são apresentados os resultados obtidos nas duas aplicações e, for fim, no capítulo 5 tem-se as considerações finais da pesquisa.

2 REVISÃO DA LITERATURA

A revisão da literatura é fundamentada em três tópicos. O primeiro trata do protocolo da pesquisa, o qual descreve a forma de seleção de trabalhos relacionados ao tema da presente pesquisa. No segundo tópico tem-se os trabalhos correlatos e, no último tópico, é apresentado o embasamento teórico de temas relacionados com o estudo desta pesquisa.

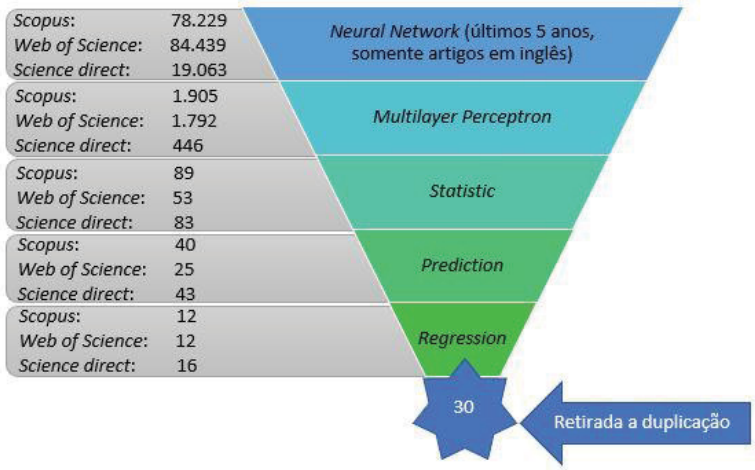
2.1 PROTOCOLO DA PESQUISA

A fim de buscar trabalhos correlatos com o tema da presente pesquisa, foi feita uma seleção de artigos em três bases de dados: *Scopus*, *Web of Science* e *Science Direct*.

Com a intenção de obter os trabalhos relacionados com o assunto nos anos mais recentes, o período considerado para a seleção foi de 2016 até 2020, considerando somente artigos em inglês.

A partir disto foram utilizadas palavras de busca, todas consideradas no resumo, título e palavras-chave do artigo. Inicialmente foi feito o filtro com a palavra *Neural Network* e, como pode-se verificar na Figura 1, obteve-se um volume considerável de artigos nas três bases. Posteriormente, foram considerados os artigos que contivessem a palavra *Multilayer Perceptron,* rede na qual as aplicações da presente pesquisa foram realizadas.

FIGURA 1 - FILTROS DA PESQUISA



FONTE: A autora (2020).

Com os filtros seguintes: *Statistic*, *Prediction* e *Regression* obteve-se uma redução considerável no volume de artigos, o que traz indícios de que o tema escolhido não possui grande volume de publicações.

Ao final, foram retirados os artigos duplicados nas três bases, o que resultou na seleção de 30 artigos com temas relacionados ao da pesquisa, os quais são descritos na seção seguinte.

2.2 TRABALHOS CORRELATOS

Nesta seção é realizada uma breve descrição dos trabalhos correlatos resultantes dos filtros de pesquisa. No geral, são aplicadas além das redes neurais MLP outras estruturas de rede e a comparação é realizada com modelos como o de regressão. Além disso, a estatística é utilizada para a comparação entre os modelos. Entre os artigos selecionados, não houve um que propusesse uma relação de rede neural como componente de um modelo estatístico de regressão.

Xu et al. (2019) desenvolveram um modelo de aprendizado profundo para a predição de transtorno depressivo em idosos nos Estados Unidos. O modelo proposto utiliza uma estrutura de rede neural recorrente com uma estrutura multitarefa.

Segura-Bedmar et al. (2018) exploraram diferentes técnicas de aprendizado de máquina para processar grandes dados do RME (Registros Médicos Eletrônicos), a fim de reduzir os esforços necessários para a realização de estudos epidemiológicos sobre anafilaxia. Além dos algoritmos clássicos de aprendizado de máquina, também foi utilizada a rede neural convolucional.

Obrzut et al. (2017) avaliaram a utilidade de modelos de inteligência artificial para previsão de sobrevida global em cinco anos em pacientes com câncer cervical tratados por histerectomia radical. Utilizaram rede neural probabilística, rede *Multilayer Perceptron* (MLP), classificador de programação de expressão gênica, algoritmo de máquinas de vetores de suporte, rede neural de função de base radial e algoritmo *k-Means*. Os resultados dos métodos de inteligência computacional foram comparados com os resultados da análise de regressão linear.

Wei (2016) examinou várias técnicas baseadas em regressão e uma rede neural artificial é utilizada para previsão de vazões durante tufões.

Kabeshova et al. (2016) compararam critérios de desempenho de modelos estatísticos lineares e não lineares para o risco de queda em moradores mais velhos de uma comunidade.

Costache e Bui (2019) popuzeram conjuntos de estatísticas bivariadas e inteligências artificiais para prever a suscetibilidade a inundações. Seis modelos foram propostos, sendo eles: rede neural MLP com razão de freqüência, rede neural MLP com pesos de evidência, taxa de freqüência florestal de rotação, rotação pesos florestais de evidência, taxa de frequência de árvore de classificação e regressão e pesos de evidência de classificação e regressão de árvore.

Bui et al. (2019) propuzeram uma nova abordagem de computação macia que é a integração de uma *Extreme Learning Machine* (ELM) e uma otimização de enxame de partículas para a previsão espacial de inundações repentinas. A comparação foi feita com três técnicas: redes neurais MLP, máquina de vetores de suporte e árvore de decisão.

O objetivo da pesquisa de Asadzadeh et al. (2019) foi estudar a eficiência das funções de pedotransferência e redes neurais artificiais para a previsão da capacidade de troca catiônica usando propriedades de solo disponíveis.

Senyurp e Ercanli (2019) utilizaram os modelos da rede neural *Multilayer Perceptron* e função de base radial para prever volumes individuais de árvores de pinheiros da Crimeia nas florestas de Cankiri e utilizaram estatísticas para comparação entre os modelos.

Mollalo et al. (2019) utilizaram RNAs para modelar a distribuição geográfica da tuberculose nos EUA. O padrão espacial da distribuição da doença foi avaliado por meio de estatísticas globais e locais. Então, foi investigada a aplicabilidade da RNA *Multilayer Perceptron* para prever a incidência da doença.

Nguyen et al. (2018) fizeram um estudo para analisar o padrão espacial de risco de incêndio para a floresta tropical do distrito de Thuan Chau usando algoritmos avançados de aprendizado de máquina, classificador de máquina de vetores de suporte, florestas aleatórias e a rede neural *Multilayer Perceptron*. Testes estatísticos foram utilizados para a comparação.

Yakubu et al. (2018) realizaram uma previsão para o índice de estresse térmico em galinhas poedeiras. Os dados foram analisados por meio de procedimentos de modelagem linear automática e de rede neural artificial.

Yu et al. (2018) desenvolveram uma estrutura de previsão de espectro com uma abordagem de aprendizado profundo em dois conjuntos de dados. A rede neural de memória de longo prazo na aprendizagem profunda foi avaliada e comparada à *Multilayer Perceptron.*

Hussain e AlAlili (2017) apresentaram quatro arquiteturas diferentes da RNA: MLP, sistema de inferência neuro-fuzzy adaptável, rede neural exógena recorrente autorregressiva não-linear e redes neurais de regressão generalizada. Uma análise multirresolução *wavelet* é aplicada para decompor os sinais meteorológicos complexos. Para a validação são utilizadas métricas de estatística.

Yu et al. (2017) abordam a avaliação da nitidez da imagem cega usando uma rede neural convolucional rasa. A rede utiliza uma camada de recurso único para descobrir recursos intrínsecos para a representação da nitidez da imagem e utiliza a MLP para avaliar a qualidade da imagem. Utiliza também rede neural de regressão geral e a regressão de vetores.

Loutfi et al. (2017) apresentaram uma comparação entre MLP e o modelo neural autorregressivo com entradas exógenas para gerar radiação solar global horária na cidade de Fes, em Marrocos.

Belciug e Sandita (2017) realizaram uma previsão do preço de fechamento diário de dez empresas listadas na bolsa de valores da Romênia usando diferentes técnicas de aprendizado de máquina, como regressão linear múltipla e configuração de redes neurais *Multilayer Perceptron.*

Navas et al. (2020) exploraram a possibilidade de desenvolver um modelo de previsão da velocidade do vento por meio de rede neural de percepção em camadas múltiplas, da Rede Neural com função de base radial e da regressão categórica.

Cabaneros et al. (2017) propuseram um sistema de aviso prévio baseado em ferramentas de previsão para contornar os efeitos adversos da exposição aos principais poluentes do ar. Modelo MLP foi treinado e para a seleção das variáveis utilizaram métodos como regressão *Stepwise*, análise de componentes principais e árvores de classificação.

Heidari et al. (2016) fizeram uma previsão para a viscosidade dos nanofluidos, a qual foi realizada por meio de uma rede neural *Multilayer Perceptron* com o algoritmo de treinamento *Levenberg-Marquardt*.

Mirbagheri e Mohammadi (2019) realizaram uma previsão de efeitos ambientais na intensidade do sinal recebido na estação de FM / TV com base em

parâmetros meteorológicos usando rede neural artificial MLP, além de regressão linear múltipla.

Parveen et al. (2020) utilizaram técnicas de regressão de vetores de suporte, rede neural MLP, rede neural de função de base radial e redes neurais de regressão geral para prever o perfil de profundidade de sólidos que fluem em um forno rotativo.

Pino-Mejías et al. (2018) utilizaram redes neurais artificiais e modelos de previsão de regressão linear como modelos preditivos para prevenir a pobreza de combustível por meio do índice de risco potencial de pobreza de combustível.

Ashrafian et al. (2018) utilizaram cinco modelos diferentes de inteligência artificial, *Splines* de regressão adaptativa multivariada, árvore modelo M5P, máquinas de vetor de suporte do menor quadrado, rede neural MLP e regressão linear múltipla foram desenvolvidas para prever a resistência à compressão e a velocidade de pulso ultrassônico do concreto reforçado com fibra.

Khalifeh e Vaferi (2019) realizaram uma previsão dos efeitos da agregação na condutividade térmica de diferentes nanofluidos. Para isso, técnicas de redes neurais MLP e função de base radial, além da regressão generalizada e retropropagação em cascata *feedforward back* foram examinadas e seus desempenhos comparados.

Bispo et al. (2017) desenvolveram um sensor macio baseado em uma rede neural artificial para estimar a viscosidade aparente dos fluidos de perfuração à base de água. O desempenho da RNA selecionada e os modelos de regressão estatística foram comparados entre si.

Ribeiro e Coelho (2020) utilizaram séries temporais mensais referentes ao preço pago aos produtores no Paraná, Brasil, por um saco de 60 kg de soja e trigo. São aplicados os métodos florestas aleatórias, máquina de aumento de gradiente e máquina de aumento de gradiente extremo e empilhamento. As técnicas vetor de suporte para regressão, rede neural MLP e vizinhos K-mais próximos são adotados como modelos de referência.

Chen et al. (2019) realizaram uma exploração de algoritmos estatísticos de aprendizagem e conjuntos de preditores genéticos a fim de detectar a resistência aos medicamentos antituberculose. Foram aplicadas redes neurais multitarefa e de tarefa única, ampla e profunda, rede neural MLP, regressão logística regularizada e classificadores florestais aleatórios.

Ke et al. (2017) propuseram uma arquitetura hierárquica genérica. Utiliza k classes, para classificação em k subconjuntos. Três abordagens são usadas para

executar esta tarefa neste estudo: classificação rígida; meios c difusos; e algoritmos genéticos. Também utiliza máquina de vetor de suporte, modelos de regressão k e compara com regressão linear, redes neurais MLP e vetor de suporte à regressão.

Gunai et al. (2016) utilizaram regressão linear múltipla e redes neurais artificiais para a previsão da demanda bruta anual de eletricidade.

2.3 REFERENCIAL TEÓRICO

Nesta seção estão descritas as técnicas utilizadas nas aplicações da presente pesquisa.

2.3.1 REDES NEURAIS ARTIFICIAIS

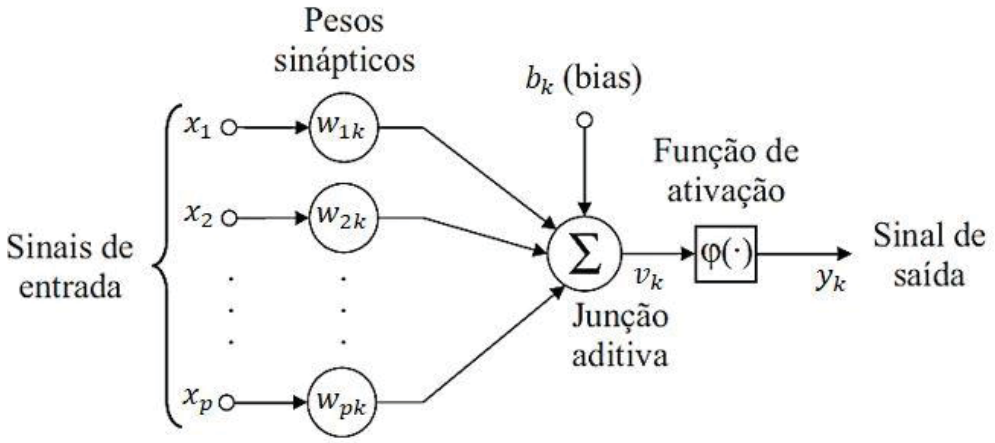
As RNAs têm a capacidade de aprendizagem, que ocorre nos processos iterativos dos ajustes dos pesos. Há diferentes tipos de aprendizagem e, portanto, não há um único algoritmo para todas as redes, a diferença é basicamente no ajuste. Os principais parâmetros que definem a arquitetura das RNAs são: número de camadas, número de neurônios em cada camada, tipo de conexão entre os neurônios e topologia da rede (BRAGA; CARVALHO; LUDEMIR, 2000).

Uma grande vantagem das RNAs é que são flexíveis e não há necessidade de muitas suposições iniciais ao modelo, além de poderem ser utilizadas em séries não lineares (KHASHEI; BIJARI, 2010).

Vários modelos já foram desenvolvidos com variação no processo de aprendizagem e da arquitetura, cada qual possui seu algoritmo de treinamento de acordo com as regras de aprendizagem adotadas, também adota uma arquitetura determinada pelo número de camadas, tipo de conexão entre os neurônios e topologia de rede (MÓDOLO, 2016).

Com a FIGURA 2, pode-se observar o funcionamento de um neurônio, em que tem-se as covariáveis da camada de entrada x_j, com $j = 1, 2, \dots, p$ e os pesos sinápticos w_{jk}, com j representando o sinal de entrada da sinapse para o neurônio k.

FIGURA 2 - MODELO DE NEURÔNIO



Fonte: Adaptada de Haykin (2001).

Na junção aditiva é feito o somatório dos sinais de entrada ponderados pelos pesos sinápticos do neurônio em questão, como pode ser observado na equação (01). Na equação (02), tem-se o acréscimo do termo *bias* (b_k) (ZANETTI et al., 2008), que pode ter valor positivo ou negativo e não faz parte das entradas recebidas pela rede, ocorre externamente. Este termo aumenta ou diminui a entrada líquida da função de ativação e, de acordo com seu valor positivo ou negativo, permite deslocar o hiperplano da origem (HAYKIN, 2001)

$$u_k = \sum_{j=1}^{p} x_j w_{jk}\,, \tag{01}$$

$$v_k = u_k + b_k\,. \tag{02}$$

O termo v_k é o potencial de ativação e $\varphi(.)$ a função de ativação, que deve ser estipulada para o problema. O sinal de saída do neurônio k é dado por y_k conforme a equação (03)

$$y_k = \varphi(v_k). \tag{03}$$

As características dos dados permitem usar diferentes tipos de funções de ativação ($\varphi(.)$) o que facilita a adequação da saída do neurônio ao tipo do problema. Na TABELA 1 estão quatro tipos de funções de ativação que podem ser utilizadas:

limiar, linear por partes, sigmóide logística e tangente hiperbólica (QUILES, 2004), sendo v a saída do combinador linear, a o parâmetro de inclinação.

TABELA 1 - EXEMPLOS DE FUNÇÃO DE ATIVAÇÃO

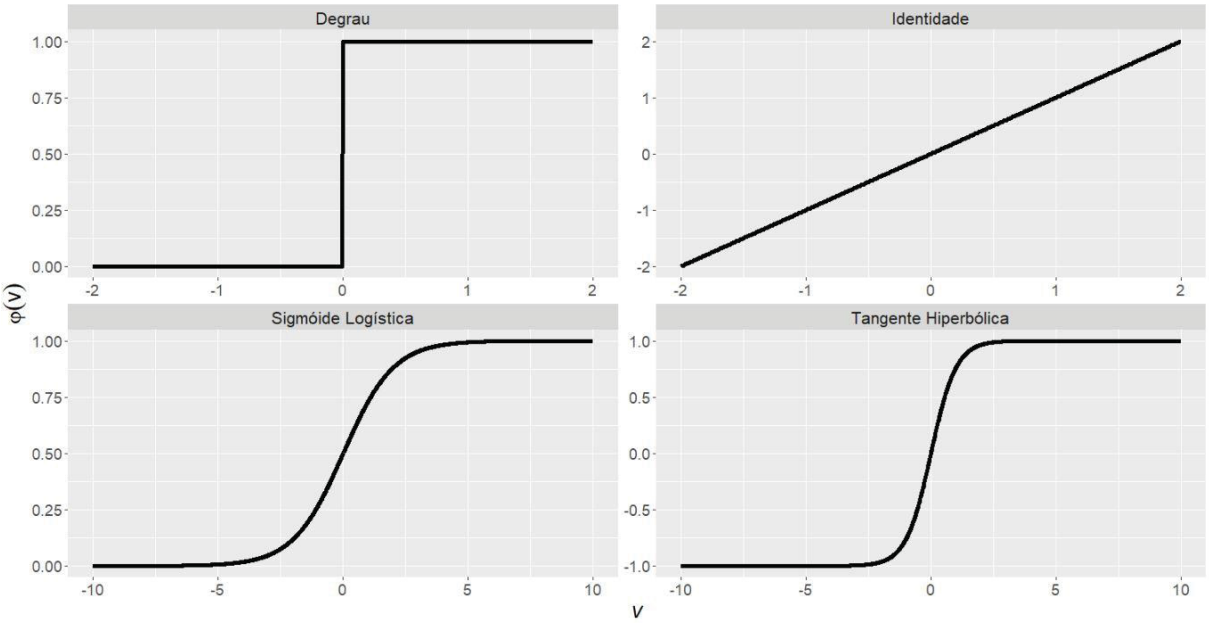
Tipo de Função	**Função**
Degrau	$\varphi(v) = \begin{cases} 1, se\ v \geq 0 \\ 0, se\ v < 0 \end{cases}$
Identidade	$\varphi(v) = v$
Sigmóide Logística	$\varphi(v) = \frac{1}{1 + e^{-av}}$
Tangente Hiperbólica	$\varphi(v) = \frac{e^{v} - e^{-v}}{e^{v} + e^{-v}}$

FONTE: Adaptado de Aggarwal (2018).

A função Degrau define a saída do neurônio para 0 se o argumento de função for menor que 0, ou 1 se seu argumento for maior ou igual a 0. A função Linear por partes é uma variação da função Limiar que cresce linearmente e permite, além da saída dos valores 0 e 1, a saída de valores reais no intervalo [0,1]. A função Sigmóide Logística também tem como saída valores reais no intervalo [0,1]. O parâmetro a permite que se obtenham diferentes inclinações. É uma das funções mais utilizadas na construção das RNAs pois exibe um balanceamento adequado entre o comportamento linear e o comportamento não linear. A função tangente hiperbólica é usada nos casos em que a saída de função deve estar no intervalo entre -1 e +1, assumindo assim uma forma antissimétrica em relação a origem (HAYKIN, 2001).

A escolha da função de ativação é uma definição importante no processo de aprendizagem das RNAs, pois ela impacta diretamente na saída da rede (resultados). Na FIGURA 3 está ilustrado o comportamento gráfico das funções de ativação exemplificadas na TABELA 1, para esta ilustração, o parâmetro a foi considerado como 1.

FIGURA 3 - EXEMPLOS GRÁFICOS DE FUNÇÃO DE ATIVAÇÃO



FONTE: A autora (2020).

2.3.1.1 Aprendizagem de uma Rede Neural Artificial

De acordo com as características do conjunto de exemplos usados no treinamento, pode-se ter uma aprendizagem supervisionada ou não supervisionada, sendo que a primeira ocorre quando cada exemplo apresenta uma saída esperada e quando não existe uma saída esperada a aprendizagem é chamada não supervisionada (BRAGA; CARVALHO; LUDEMIR, 2000).

Na aprendizagem supervisionada há um supervisor externo que fornece a saída desejada para cada entrada da rede. Deseja-se ajustar os parâmetros da rede de maneira que, para cada entrada, a saída da rede seja aquela fornecida. Cada saída da rede tem seu erro calculado, o qual é utilizado pelo algoritmo de treinamento para ajuste dos pesos sinápticos a fim de minimizar esse erro.

Os modelos de RNAs *Perceptron* e *Multilayer Perceptron* se enquadram nesta característica. No caso da aprendizagem não supervisionada, não há esta saída que acompanhe o aprendizado, o conjunto de exemplos utilizado no treinamento contém apenas os padrões de entrada, os modelos de Hopfield e o Mapa de Kohonen usam essa característica em seu algoritmo de aprendizagem. Portanto, não existe um único algoritmo de aprendizagem e basicamente as diferenças entre os tipos de aprendizagem dizem respeito a maneira como é feito o ajuste dos pesos sinápticos de um neurônio (MÓDULO, 2016).

Segundo Haykin (2001), há cinco regras básicas de aprendizagem para as RNAs. A aprendizagem por correção de erro, que se fundamenta na filtragem ótima, a aprendizagem baseada em memória trabalha memorizando os dados de treinamento, as aprendizagens hebbiana e competitiva são inspiradas em considerações neurobiológicas e a aprendizagem de Boltzmann cuja base vem da mecânica estatística.

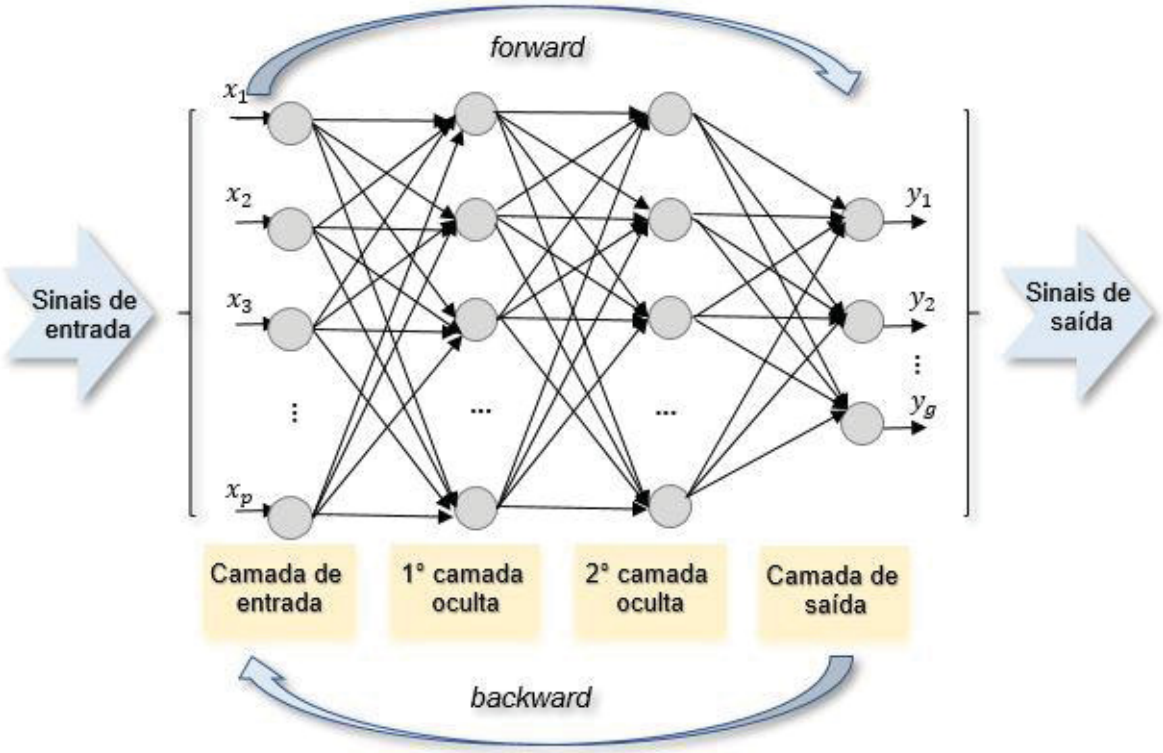
2.3.1.2 Modelo Multilayer Perceptron

Vários modelos de RNAs foram desenvolvidos com variações no processo de aprendizagem e na arquitetura da rede. Cada modelo de aprendizagem tem seu algoritmo de treinamento, de acordo com as regras de aprendizagem adotadas, e também adota uma determinada arquitetura de rede, de acordo com o número de camadas, o tipo de conexão entre os neurônios e a topologia da rede (MÓDULO, 2016).

O modelo *Multilayer Perceptron* (MLP), cuja aprendizagem é supervisionada por correção de erro, possui arquitetura com mais de uma camada, é acíclica (quando a saída de um neurônio não pode servir de entrada para algum neurônio anterior) e conectada (cada entrada é processada por todos os neurônios). A regra de propagação é dada pelo produto interno das entradas ponderadas pelos pesos com adição do termo *bias* e, a saída da camada anterior é a entrada da camada atual (HAYKIN, 2001). Uma rede MLP, nada mais é do que uma generalização da *Perceptron* de camada única (HAYKIN, 2009).

Segundo Fausett (1994), o treinamento da MLP ocorre nas etapas: *feedforward*, *backpropagation* e ajuste dos pesos. Na primeira, ocorrem os treinamentos de padrões com as entradas, na segunda a retropropagação por correção dos erros e na terceira os pesos são ajustados. Na FIGURA 4 tem-se a representação desta rede.

FIGURA 4 – REDE NEURAL *MULTILAYER PERCEPTRON*



FONTE: A autora (2020).

Os passos para o algoritmo desta rede ocorrem de acordo com a sequência dada a seguir, sendo que j refere-se aos neurônios da camada de entrada, k aos da camada oculta e h aos neurônios da camada de saída. O número da iteração foi representado por t , que inicia em 1.

Propagação *Forward:*

a) Os pesos são iniciados com valores randômicos $|w| \leq 1$ e $|b| \leq 1$;

b) São calculados os valores da função aditiva, dada pelas equações (04) e (05), para cada neurônio k da camada oculta e na sequência o valor de transferência resultante da equação (06);

$$u_k^{(t)} = \sum_{j=1}^{p} x_j^{(t)} w_{jk}^{(t)}, \quad (04)$$

$$v_k^{(t)} = u_k^{(t)} + b_k^{(t)}, \quad (05)$$

$$a_k^{(t)} = \varphi\left(v_k^{(t)}\right). \quad (06)$$

Para a aplicação deste trabalho a função de ativação não linear sigmóide logística foi utilizada na camada oculta e seus valores de entrada e saída podem ser contínuos, conforme a equação (07)

$$a_k^{(t)} = \varphi\left(v_k^{(t)}\right) = \frac{1}{1 + e^{-v_k^{(t)}}}. \quad (07)$$

c) Para cada neurônio h da camada de saída faz-se conforme as equações (08), (09) e (10), nos casos em que há mais de um neurônio nesta camada. A função de ativação para esta etapa pode ser a mesma ou diferente, neste trabalho foi utilizada a identidade

$$u_h^{(t)} = \sum_{j=1}^{n} a_k^{(t)} w_{kh}^{(t)}, \quad (08)$$

$$v_h^{(t)} = u_h^{(t)} + b_h^{(t)}, \quad (09)$$

$$a_h^{(t)} = \varphi\left(v_h^{(t)}\right). \quad (10)$$

Propagação *Backward:*

d) O erro de previsão é então calculado e um algoritmo de aprendizagem é utilizado para o ajuste dos pesos a fim de minimizar o erro. Na sessão 2.3 estão descritos os algoritmos de aprendizado contemplados no pacote *neuralnet*, o qual é utilizado para as aplicações.

e) Os pesos são atualizados e é possível seguir para a próxima iteração $t + 1$ com os novos pesos. Isso ocorre até que o critério de parada seja atendido.

2.3.2 SIMULAÇÃO DOS DADOS

Nos capítulos seguintes foram utilizados dados simulados a fim de contextualizar os métodos abordados. Para que a geração desses dados fosse próxima a de dados reais, optou-se por utilizar 5 covariáveis históricas diárias de preços do petróleo, cada qual com 1.000 observações, sendo a variável resposta a previsão para o próximo dia. Os parâmetros foram estimados por meio de uma rede neural *Multilayer Perceptron* de 2 neurônios na camada oculta e função sigmóide

logística na mesma, na camada de saída a função utilizada foi a identidade. Os valores dos 15 parâmetros estimados foram então guardados.

Para o estudo de simulação fez-se então uma só passagem com as 5 covariáveis pela equação (11), a qual consiste na fase *forward* de uma rede de mesma estrutura da que gerou os 15 parâmetros estimados e guardados anteriormente. Vale ressaltar que não foi feito treinamento nesta parte, a intenção foi somente a de gerar os dados com uma estrutura de rede conhecida, sendo assim, tem-se a informação dos parâmetros que seriam esperados ser estimados pelos algoritmos estudados

$$\mu_i = -3{,}4 + 2{,}5\frac{1}{(1+e^{-(2{,}40+1{,}20x_{i1}+0{,}04x_{i2}-0{,}02x_{i3}+0{,}03x_{i4}-0{,}02x_{i5})})} + 4{,}1\frac{1}{(1+e^{-(-1{,}10+0{,}89x_{i1}+0{,}06x_{i2}-0{,}05x_{i3}+0{,}12x_{i4}-0{,}08x_{i5})})}. \quad (11)$$

Então, considerando que um modelo estima a média dos dados, ou seja, o valor esperado, tem-se conforme a equação (11) o modelo que gerou a média μ_i, $i = 1, 2, \ldots, 1.000$ dos dados simulados.

Ao realizar uma predição para a variável resposta espera-se errar tanto para mais quanto para menos e portanto é razoável esperar que a resposta das predições para este modelo siga uma distribuição normal com média μ_i e uma determinada variância, que para esta simulação foi estipulada em 0,1. Para inserir a aleatoriedade, a geração dos dados foram finalizadas com $y_i = N(\mu_i, 0{,}1)$.

Sendo assim, tem-se os parâmetros que geraram a média dos dados simulados e a partir deste conhecimento deseja-se avaliar as técnica de aprendizado da rede.

2.3.3 ESTIMAÇÃO E INFERÊNCIA

A abordagem estatística para análise e resumo de informações contidas em um conjunto de dados consiste na suposição de que existe um mecanismo estocástico que gera este processo. Este mecanismo é descrito por meio de um modelo probabilístico, o qual é representado por uma distribuição de probabilidade. Na maior parte das vezes a distribuição de probabilidade geradora do processo é desconhecida, sendo assim, distribuições de probabilidade adequadas devem ser escolhidas de

acordo com o tipo de fenômeno em análise, na maioria das situações, assume-se que a distribuição de probabilidade é conhecida (BONAT, 2012).

Quando uma amostragem é realizada a partir de uma população descrita pela função densidade de probabilidade, nos casos contínuos, ou função de probabilidade nos discretos, o conhecimento do parâmetro θ gera o da população inteira. Sendo assim, é natural buscar um método para encontrar um bom estimador para θ, ou seja, um bom estimador pontual. Um estimador pontual é qualquer função $H(Y_1, \dots, Y_n)$ de uma amostra (CASELLA, 2010).

O processo de inferência consiste em encontrar um valor que seja mais plausível para os parâmetros do modelo considerando o espaço paramétrico do mesmo e assim pode-se também realizar previsões a respeito de valores futuros (BONAT, 2012). Então, é útil utilizar algumas técnicas que forneçam candidatos razoáveis para se considerar como estimador, como método dos momentos, método da máxima verossimilhança e estimadores de Bayes (CASELLA, 2010).

O método de máxima verossimilhança é a técnica mais popular para a obtenção de estimadores. Sendo $Y_1, \dots, Y_n$ uma amostra independente e identicamente distribuída de uma população com $f(y|\theta_1, \dots, \theta_k)$, então a função de verossimilhança é dada pela equação (12). O estimador de máxima verossimilhança (EMV) é o valor do parâmetro para o qual a amostra observada é mais provável. Há dois incovenientes associados ao problema geral de encontrar o máximo de uma função, sendo que o primeiro refere-se a encontrar o máximo global e verificar que realmente foi encontrado e o outro é a sensibilidade da estimativa para pequenas mudanças (BONAT, 2012; CASELLA, 2010)

$$L(\theta|y) = L(\theta_1, \dots, \theta_k|y_1, \dots, y_n) = \prod_{i=1}^{n} f(y_i|\, \theta_1, \dots, \theta_k). \qquad (12)$$

A função de verossimilhança é dada pela expressão da distribuição conjunta de todas as variáveis aleatórias envolvidas no modelo vista como função dos parâmetros, uma vez que os dados são quantidades fixas observadas. Por questões computacionais é comum utilizar a função log-verossimilhança $l(\theta|y)$ em vez de $L(\theta|y)$, já ambas trazem os mesmos resultados (BONAT, 2012).

Se a função for diferenciável em w_i então os possíveis candidatos para o EMV serão os valores que resolvem $\frac{\partial\, l(\theta|y)}{\partial \theta_i} = 0$ (CASELLA, 2010).

Assumindo que um conjunto de variáveis aleatórias $Y_1, Y_2, \dots, Y_n$ são independentes e identicamente distribuídas seguindo uma distribuição Normal de média μ e variância σ^2, então $Y_i \sim N(\mu, \sigma^2)$, conforme a equação (13) e cuja equação $\frac{\partial\, l(\theta|y)}{\partial \theta}$ se reduz a (14)

$$f(y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{y_i-\mu}{\sigma}\right)^2}, -\infty < y < \infty, \tag{13}$$

$$\frac{\partial\, l(\mu|y)}{\partial \mu} = \sum_{i=1}^{n} \frac{(y_i - \mu)}{\sigma^2}. \tag{14}$$

Considerando que para a presente aplicação tem-se um vetor de médias, para cada resposta conforme (15), então a distribuição correspondente é representada em (16) e a equação $\frac{\partial\, l(\mu|Y)}{\partial w}$ em (17).

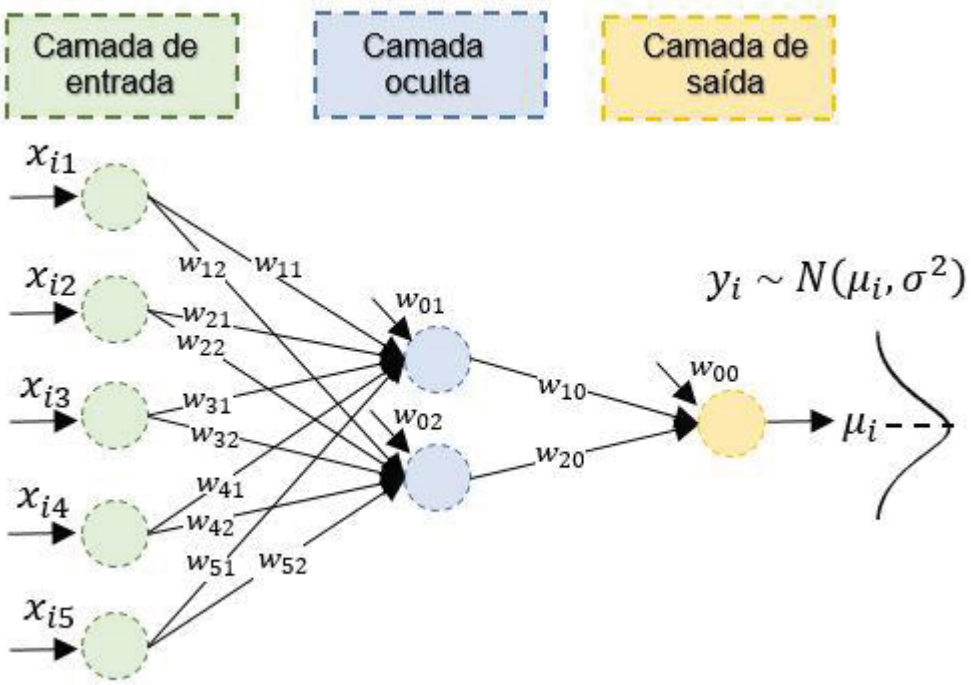
$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \quad \boldsymbol{\mu} = \begin{bmatrix} \mu_{y_1} \\ \vdots \\ \mu_{y_n} \end{bmatrix} \tag{15}$$

$$f(Y) = (2\pi)^{-\frac{p}{2}} |\Sigma|^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})' \Sigma^{-1} (\boldsymbol{y}-\boldsymbol{\mu})}, -\infty < y_i < \infty, i = 1, \dots, p \tag{16}$$

$$\frac{\partial\, l(\mu|Y)}{\partial w} = \frac{\partial\, l(\mu|Y)}{\partial w} \Sigma^{-1} (\boldsymbol{y} - \boldsymbol{\mu}) \tag{17}$$

Então, no exemplo de simulação que está sendo tratado, tem-se a arquitetura da rede conforme a FIGURA 5.

FIGURA 5 –REDE NEURAL COM SAÍDA DE DISTRIBUIÇÃO NORMAL



FONTE: A autora (2020).

Portanto, é razoável supor que a saída da rede segue uma distribuição normal cuja média consiste na equação da rede neural, conforme em (18) e (19) e tem-se o resultado de $\frac{\partial\, l(\mu|Y)}{\partial w}$ em (20).

$$f(y) = (2\pi)^{-\frac{p}{2}} |\Sigma|^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{y}-\tilde{X}w_{k0})' \Sigma^{-1}(\boldsymbol{y}-\tilde{X}w_{k0})}, -\infty < y_i < \infty, i = 1, \dots, n \tag{18}$$

$$\mu = \tilde{X}w_{k0} = \begin{bmatrix} 1 & \frac{1}{1+e^{-(w_{01}+Xw_{.1})}} & \dots & \frac{1}{1+e^{-(w_{0m}+Xw_{.m})}} \end{bmatrix} \begin{bmatrix} w_{00} \\ w_{10} \\ \dots \\ w_{m0} \end{bmatrix} \tag{19}$$

$$\frac{\partial\, l(\mu|Y)}{\partial \mu} = \frac{\partial\, l(\mu|Y)}{\partial w} \Sigma^{-1}(\boldsymbol{y} - \tilde{X}w_{k0}) \tag{20}$$

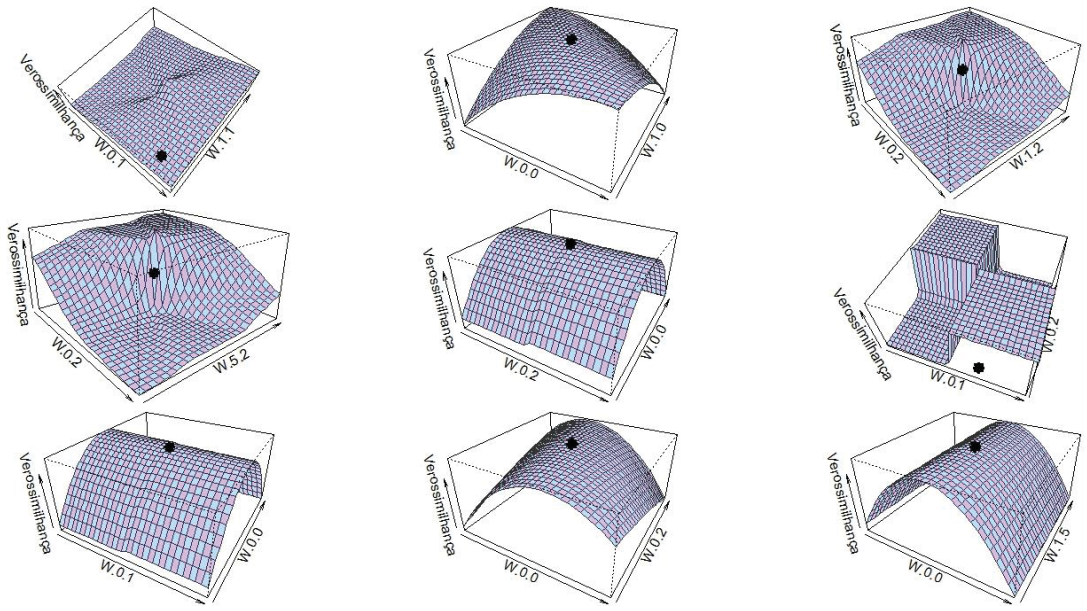
O vetor de parâmetros $\underline{W} = (w_{k0}, (w_{.1}, \dots, w_{.m}))$ que deseja-se estimar constitui dos parâmetros w_{k0} que ligam os k neurônios da camada oculta com o de saída, além do termo *bias*, e os vetores $w_{.k}$, $k = 1, \dots, m$, os quais relacionam as variáveis de entrada com o respectivo neurônio k, além do termo *bias*. Para este tipo de estrutura, quando busca-se a verossimilhança é necessário fazer a regra da cadeia $\frac{\partial l(\mu|y)}{\partial \mu} \frac{\partial \mu}{\partial \underline{W}}$.

O vetor gradiente, que neste caso é denominado vetor escore, corresponde a primeira derivada da log-verossimilhança. A segunda derivada corresponde a matriz de informação de Fischer, $I(\theta) = E\left[\left(\frac{\partial L}{\partial \theta}\right)^2\right] = -E\left[\left(\frac{\partial^2 L}{\partial \theta^2}\right)\right]$. Tem-se que o parâmetro tem distribuição assintoticamente normal $\hat{\theta} \sim N(\theta, I^{-1}(\theta))$. Assim, os valores altos de $L(\theta)$ indicam que pequenas alterações em θ resultam em grandes alterações, e $I(\theta)$ fornece a informação sobre a curvatura do logaritmo da verossimilhança, sendo que quanto mais côncava mais fácil encontrar o $\hat{\theta}$ máximo.

No caso dos dados simulados, tem-se o vetor $\underline{W} = (w_{k0}, w_{.1}, w_{.2})$. Para encontrar o ótimo da função pode-se buscar analiticamente resolvendo $\frac{\partial\, l(\mu|y)}{\partial \mu}\frac{\partial \mu}{\partial \underline{W}} = 0$, o que corresponde a função escore, ou numericamente por meio de otimizações com algoritmos, assim obtendo aproximações numéricas. Há ainda outras formas como por exemplo simulações, bastante usual em Inferência Bayesiana e aproximações por verossimilhança, denominadas pseudo-verossimilhança.

Na FIGURA 6 é possível verificar graficamente a função de log-verossimilhança para os dados simulados. Os parâmetros são avaliados em pares e os demais são fixados no valor que gerou a média dos dados. Seria necessário gerar $C_{15,2} = 105$ gráficos para verificar todos os parâmetros, porém, com 9 já é possível avaliar a característica da função de verossimilhança e a dificuldade na busca pelo ótimo.

FIGURA 6 - VEROSSIMILHANÇA DOS DADOS SIMULADOS



FONTE: A autora (2020).

Os pontos pretos na FIGURA 6 estão posicionados nas coordenadas em que a média dos dados foi gerada. É possível perceber que estão em uma região do gráfico que corresponde à de maior plausibilidade para o valor do parâmetro, porém, também é possível notar que há outros valores de parâmetros que assim como este correspondem a mesma plausibilidade.

No contexto de otimização de parâmetros via redes neurais, há alguns pacotes no *software* R que realizam tal processo, como o *neuralnet*. A função *neuralnet()* deste pacote possui alguns algoritmos implementados para o aprendizado: *backprop*, *rprop+*, *rprop-*, *sag* e *slr*.

Para a função *neuralnet()*, os valores iniciais dos parâmetros são gerados de forma aleatória de uma distribuição normal padrão e o processo ocorre em alguns passos. Primeiramente, a rede neural calcula uma saída $\hat{y}$ dadas as entradas, os parâmetros iniciais e a arquitetura da rede. Então, a função de perda é avaliada, sendo esta a soma dos quadrados dos erros (E), dada em (21), no caso da saída estudada (distribuição normal), com $l = 1, \dots, L$ correspondendo às observações. Então, os parâmetros são adaptados conforme a regra do algoritmo de aprendizado, este processo ocorre até que um critério de parada seja atingido, que pode ser o número de iterações, por exemplo (GÜNTHER; FRITSCH, 2010)

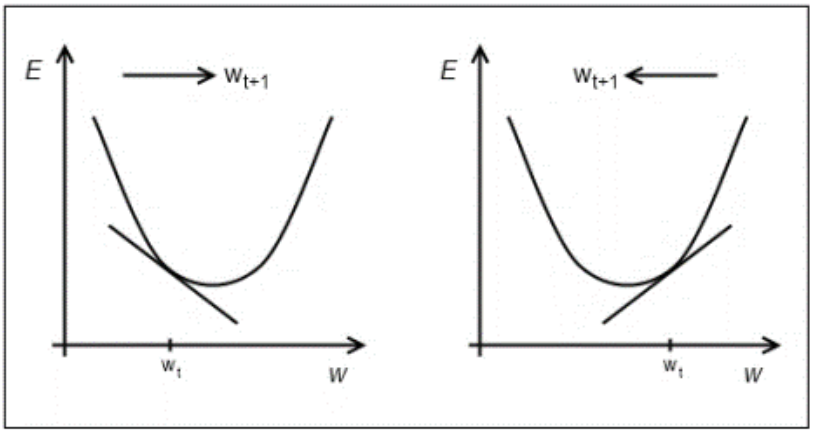
$$E = \frac{1}{2}\sum_{l=1}^{L}(\hat{y}_l - y_l)^2. \tag{21}$$

$$\hat{y}_l = \tilde{X} w_{k0}$$

Os algoritmos de aprendizado da função *neuralnet()* buscam minimizar o valor da função de erro. Os métodos de gradiente descendente são bastante utilizados para o aprendizado supervisionado das redes neurais, sendo o mais popular o *backpropagation* (RUMELHART; MCCLELLEND,1986; ANASTASIADIS *et al.*, 2005), que no caso da função *neuralnet()* corresponde ao *backprop*.

Neste algoritmo, sendo t *a* indexação das etapas de iteração e w os parâmetros, o gradiente da função de erro $\Delta w^{(t)} = \frac{\partial E(w^{(t)})}{\partial w}$ é calculado em relação aos parâmetros para assim encontrar a raiz. Então, os parâmetros são modificados a cada iteração. A direção é oposta as derivadas parciais, conforme FIGURA 7, isto ocorre até que um mínimo local seja atingido.

FIGURA 7 – ALGORITMO DE *BACKPROPAGATION*



FONTE: Günther e Fritsch (2010).

Caso a derivada parcial seja negativa, o peso é aumentado assim como se for positiva, o peso é reduzido, e assim um mínimo local é alcançado. Todas as derivadas parciais são calculadas usando a regra da cadeia, dado que a função calculada de uma rede neural é basicamente uma composição de integração e funções de ativação. Na equação (22) tem-se o algoritmo denominado *backpropagation* (GÜNTHER; FRITSCH, 2010)

$$w^{(t+1)} = w^{(t)} - \eta \Delta w^{(t)}. \tag{22}$$

O algoritmo busca minimizar a função de erro adicionando uma taxa de aprendizado $0 \leq \eta \leq 1$ aos parâmetros na direção oposta à do gradiente. Alguns algoritmos adaptativos baseados no gradiente buscaram superar a dificuldade em encontrar as taxas de aprendizado corretas para cada região do espaço de busca. Um algoritmo que obteve destaque foi o de *Resilient backpropagation* (*Rprop*), introduzido por Riedmiller e Braun (ANASTASIADIS *et al.*, 2005; RIEDMILLER; BRAUN, 1993).

As estratégias de aprendizado podem ser de adaptação global ou local, no sentido de que na primeira fazem uso da rede inteira e modificam parâmetros globais, já nas locais modificam cada parâmetro especificamente. O *Rprop* é baseado no algoritmo de *backpropagation*, porém, a taxa de aprendizagem η_k é separada para cada parâmetro e pode ser alterada no processo de treinamento, sendo assim, não há uma definição de uma taxa geral de aprendizado. Além disso, em vez da magnitude

das derivadas parciais, apenas seu sinal ($sign$) é usado para atualizar os pesos, os parâmetros são ajustados pela regra conforme em (23) (RIEDMILLER; BRAUN, 1993)

$$w_k^{(t+1)} = w_k^{(t)} - \eta_k^{(t)} sign\left(\Delta w_k^{(t)}\right). \tag{23}$$

Com o intuito de acelerar a convergência em áreas pouco profundas, a taxa de aprendizado η_k é aumentada se a derivada parcial correspondente mantiver seu sinal. Assim como será diminuída se o sinal da derivada parcial da função de erro for alterada, pois um sinal de mudança indica que o mínimo foi perdido devido a uma taxa de aprendizado muito elevada. Há dois algoritmos de *Rprop* disponíveis na função *neuralnet()*, sendo com (*rprop+*) e sem (*rprop-*) retrocesso do parâmetro. O retrocesso do parâmetro é uma técnica para desfazer a última iteração e adicionar um valor menor para o parâmetro na próxima etapa. Sem o uso desta técnica, o algoritmo pode pular o mínimo diversas vezes (GÜNTHER; FRITSCH, 2010).

Após *Rprop*, foram construídas variações, como o algoritmo globalmente convergente modificado. Consiste no *Rprop* com uma modificação na taxa de aprendizado, que é alterada conforme a equação (24), com $0 < \delta \ll \infty$. A taxa de aprendizado é calculada de acordo com a menor derivada parcial absoluta ou a menor taxa de aprendizado i , os algoritmos *slr* e *sag* da função *neuralnet()* correspondem a esta abordagem (ANASTASIADIS *et al.*, 2005)

$$\eta_i^{(t)} = - \frac{\sum_{k;i\neq k} \eta_k^{(t)} \Delta w_k^{(t)} + \delta}{\Delta w_i^{(t)}}. \tag{24}$$

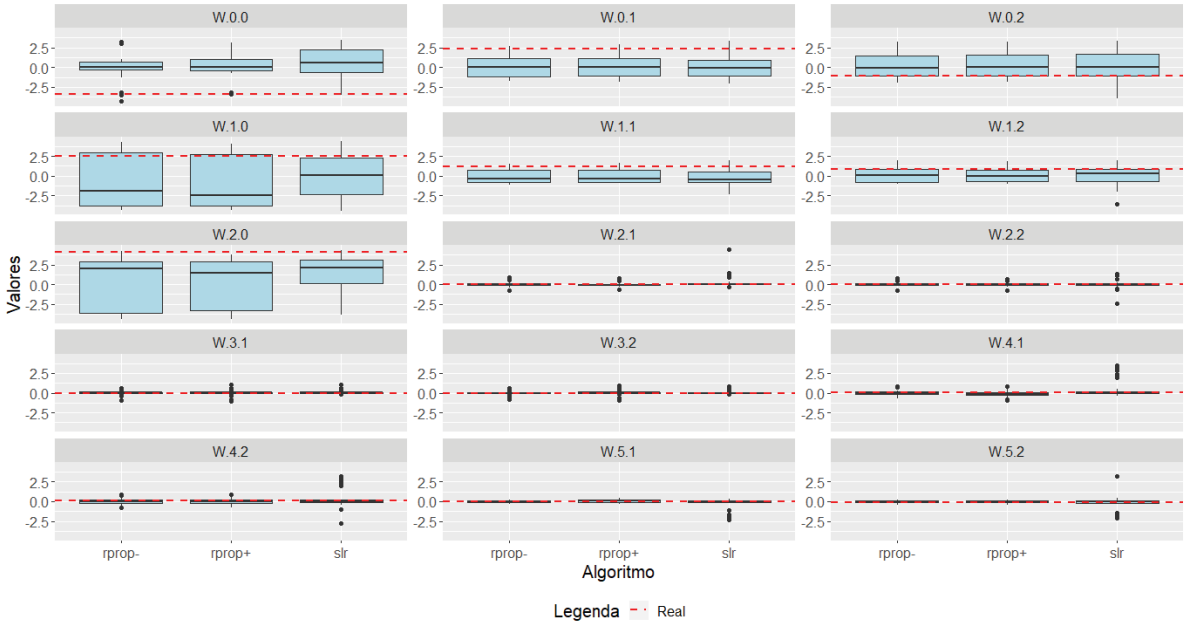
Os 5 algoritmos de aprendizado da função *neuralnet()* foram então aplicados nos dados simulados considerando sempre as 5 entradas com 2 neurônios na camada oculta, perfil com o qual os dados simulados foram gerados. Assim, pode-se avaliar se os algoritmos conseguiriam aproximar-se dos parâmetros geradores da média dos dados.

Para o treinamento da rede foram utilizadas as características padrões da função *neuralnet()*, então, como critério de parada o limite para as derivadas parciais da função de erro foi de 0,01 e a função de ativação da camada oculta sigmóide logística. Já o número máximo de iterações foi alterado para 100.000.000 em todos os

algoritmos devido a dificuldade dos *backprop*, *slr* e *sag* convergirem em um volume menor de iterações em testes iniciais.

Foram feitas 50 estimações dos parâmetros em cada algoritmo. Na FIGURA 8 é possível visualizar a distribuição dos 15 parâmetros estimados pelos algoritmos *rprop-, rprop+* e *slr.* Os algoritmos *backprop* e *sag* convergiram poucas vezes, sendo assim, não obtiveram as 50 convergências e não foram utilizados nas análises.

FIGURA 8 – VALORES DE PARÂMETROS ESTIMADOS PELOS ALGORITMOS



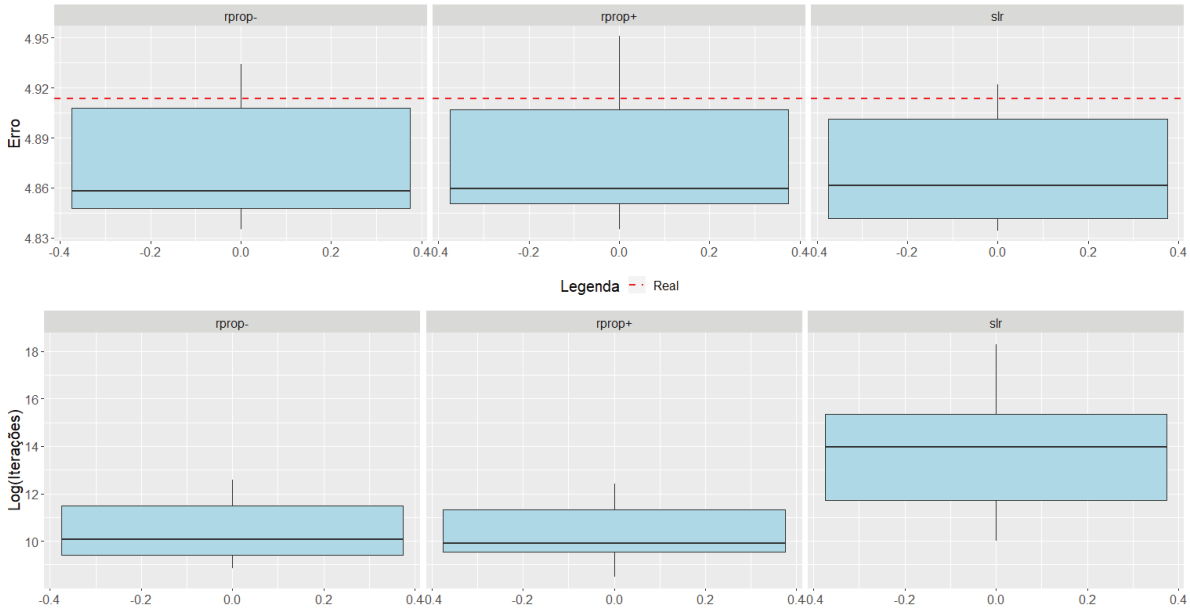
FONTE: A autora (2020).

No geral, valores discrepantes ocorreram nos parâmetros de intercepto w_{00}, w_{01} e w_{02} e naqueles correspondentes a ligação dos neurônios da camada oculta e a de saída, como w_{10} e w_{20}. A linha tracejada indicando o valor gerador da média dos dados não corresponde a média dos valores encontrados nas 50 estimações, o que indica que o valor encontrado nas estimativas não corresponde à geradora dos dados.

Nos apêndices 1 a 4 tem-se diagramas de dispersão, em que é possível verificar se há relação entre as estimações de cada um dos três algoritmos. É possível perceber que há uma relação entre as estimativas encontradas pelos algoritmos *rprop+* e *rprop-.* Já os valores estimados pelos algoritmos *rprop-* e *slr* foram bem diferentes. Os algoritmos *slr* e *rprop+* obtiveram valores estimados diferentes para os parâmetros também.

Na FIGURA 9 tem-se a distribuição dos erros no treinamento com as 50 estimações dos três algoritmos assim como o volume de iterações. Pode-se perceber que o algoritmo *slr,* no geral, precisou de mais iterações que os demais algoritmos, em contrapartida, obteve menor erro. Tem-se também que na maior parte das estimativas, os algoritmos obtiveram menor erro do que o modelo constituído pelos parâmetros que geraram a média dos dados, esse obteve erro de 4,91 e está sinalizado em linha tracejada na figura.

FIGURA 9 - ERROS E ITERAÇÕES DOS ALGORITMOS NO TREINAMENTO



FONTE: A autora (2020).

Nos 50 treinamentos realizados com o algoritmo *rprop+*, o menor erro foi de aproximadamente 4,84, com 20.278 iterações. Nos treinamentos realizados com *rprop-* tem-se que o menor erro foi de aproximadamente 4,84 também, obtido em 12.094 iterações. No algoritmo *slr* foi de 4,83 com 3.190.799 iterações. Os parâmetros obtidos com menor erro estão contidos na TABELA 2, onde fica evidente que os algoritmos *rprop-* e *rprop+* obtiveram resultados semelhantes para a estimação e diferentes do encontrado pelo algoritmo *slr.*

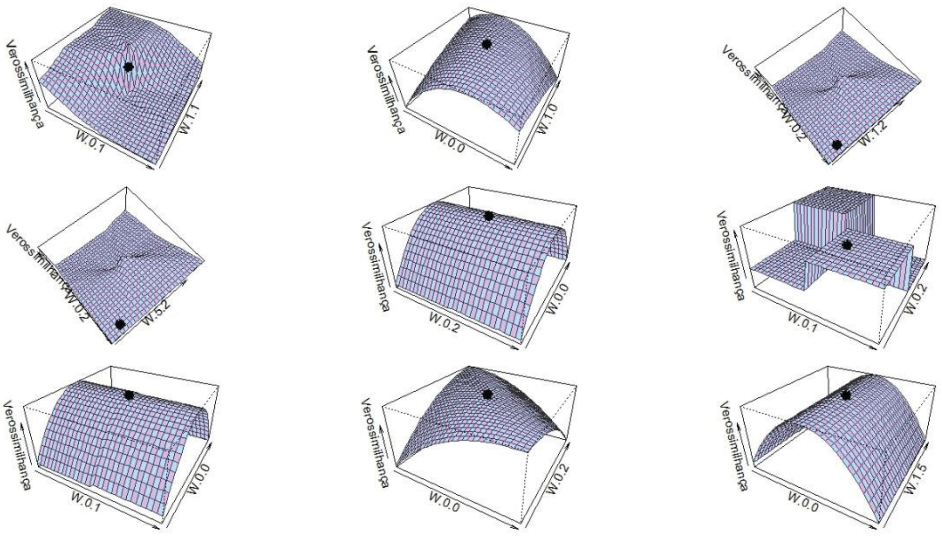
TABELA 2 – PARÂMETROS ESTIMADOS QUE RESULTARAM EM UM MENOR ERRO

Parâmetro	***rprop+***	***rprop-***	***slr***	**Real**
W_{01}	-0.70	-0.69	3.32	2.4
W_{11}	0.67	0.67	1.99	1.2
W_{21}	0.09	0.09	-0.18	0.04
W_{31}	-0.02	-0.02	-0.12	-0.02
W_{41}	0.15	0.15	-0.35	0.03
W_{51}	-0.12	-0.12	0.27	-0.02
W_{02}	3.19	3.24	0.64	-1.10
W_{12}	1.89	1.93	-0.63	0.89
W_{22}	-1.12	-0.11	-0.09	0.06
W_{32}	-0.13	-0.14	0.02	-0.05
W_{42}	-0.29	-0.29	-0.14	2.5
W_{52}	0.24	0.24	0.11	-0.08
W_{00}	-3.21	-3.19	2.28	-3.4
W_{10}	5.19	5.21	1.32	2.5
W_{20}	1.46	1.43	-5.55	4.1
Iterações	20.278	12.094	3.190.799	-

FONTE: A autora (2020).

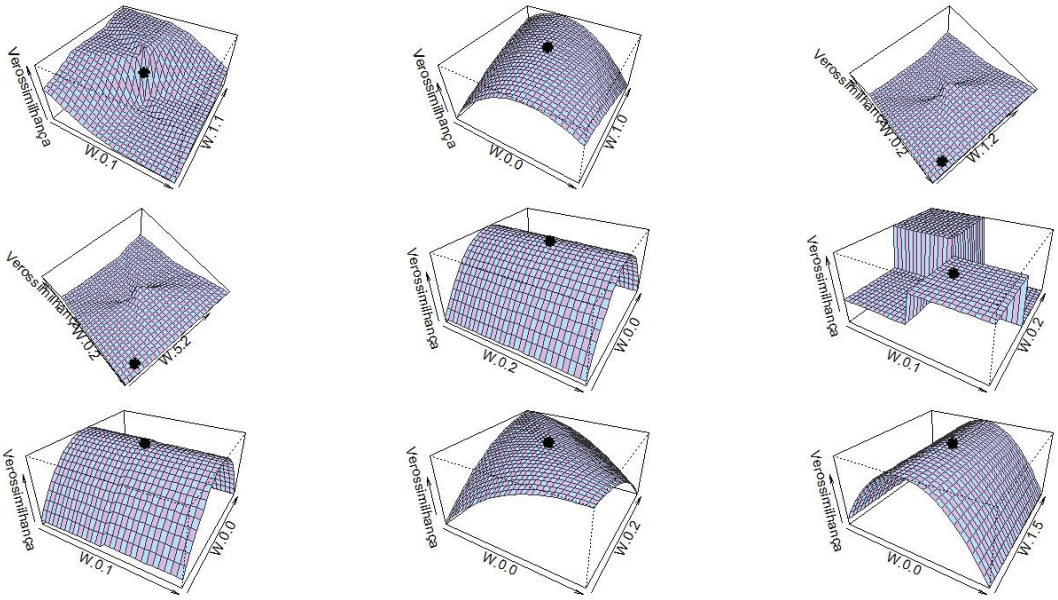
Ainda com os resultados da TABELA 2, alguns pares de parâmetros foram avaliados e os demais fixados e nas FIGURAS 10 a 12 tem-se a função de verossimilhança para cada um dos algoritmos e os pontos em preto correspondem ao valor estimado pelo algoritmo. É possível avaliar que não há um único ótimo na função.

FIGURA 10 - VEROSSIMILHANÇA *RPROP+*



FONTE: A autora (2020).

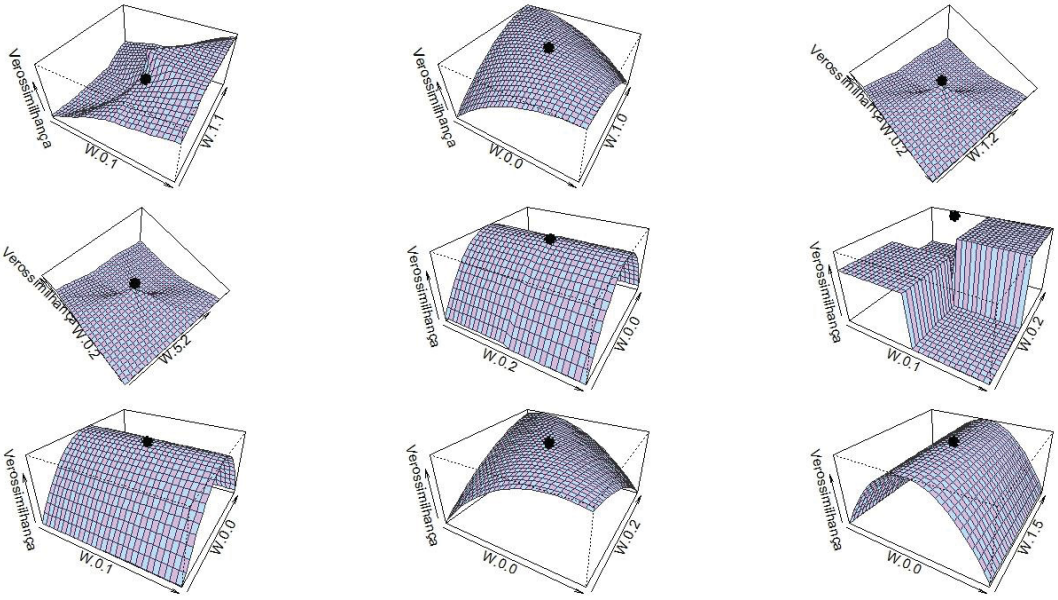
FIGURA 11 - VEROSSIMILHANÇA *RPROP* -



FONTE: A autora (2020).

É possível perceber que as diferenças entre *rprop-* e *rprop+* são bastante sutis, quase imperceptíveis, devido a similaridade nos resultados encontrados. Já para o algoritmo *slr* tem-se resultados diferentes, porém, todas refletem que não haveria um único ponto de ótimo na função.

FIGURA 12 - VEROSSIMILHANÇA *SLR*



FONTE: A autora (2020).

2.3.4 INFERÊNCIA BAYESIANA

No contexto estatístico, fazer inferências a respeito de novos fenômenos ou novas observações é um objetivo bastante comum, e há diferentes correntes para a realização deste objetivo. A corrente clássica, cujos fundadores são Karl Pearson, Ronald A. Fisher e Jerzy Neyman, foi predominante por muitos anos. Basicamente, tal corrente é fundamentada pela ideia da inferência como generalização sobre a população com base em uma amostra que foi retirada da mesma. Um aspecto importante é reconhecer a variabilidade que há de amostra para amostra e analisar os dados observados como um de muitos conjuntos que poderiam ter sido obtidos nas mesmas circunstâncias. A interpretação depende não somente da amostra observada mas também das hipóteses adotadas acerca dos possíveis conjuntos amostrais (PAULINO *et al.*, 2018).

Além da clássica, há outras correntes, como Bayesiana, estruturalista, verossimilhancista e outras mais. No contexto da inferência Bayesiana, tem-se que o ajuste de um conjunto de dados a um modelo probabilístico é resumido pelo resultado da distribuição de probabilidade dos parâmetros do modelo e as observações amostrais. As conclusões estatísticas sobre determinado parâmetro θ ou sobre dados não observados $\tilde{y}$ são realizadas com base em afirmações de probabilidades, que são condicionais ao valor observado y e podem ser escritas como $p(\theta|y)$ ou $p(\tilde{y}\,|y)$. Para fazer afirmações de θ dado y inicia-se com uma distribuição de probabilidade conjunta de θ e y . A densidade de probabilidade pode ser escrita como o produto de duas densidades que são denominadas distribuição *a priori* $p(\theta)$ e distribuição da amostra $p(y|\theta)$, tem-se então a equação (25) (GELMAN et al., 2014)

$$p(\theta, y) = p(\theta)p(y|\theta). \tag{25}$$

Condicionando o valor conhecido de y e utilizando a propriedade da probabilidade condicional tem-se a densidade *a posteriori*, descrita na equação (26). Uma forma equivalente omite $p(y)$, sendo este fixo produz a densidade *a posteriori* conforme a equação (27)

$$p(\theta|y) = \frac{p(\theta)\,p(y|\theta)}{p(y)}\,, \tag{26}$$

$$p(\theta|y) \propto p(\theta)p(y|\theta). \quad (27)$$

Portanto, na inferência Bayesiana, após a informação adicional de um acontecimento realizado, como a observação de um conjunto de dados, as probabilidades *a priori* são revistas por meio da fórmula de Bayes, dada em (26) e as probabilidades *a posteriori* são então atribuídas (PAULINO *et al.*, 2018).

Diversos métodos foram criados para a construção de amostras no contexto Bayesiano. A simulação de cadeia de Markov, ou Monte Carlo via cadeia de Markov (MCMC), é um método geral baseado nos valores obtidos de θ de distribuições aproximadas e que, em seguida, são corrigidos para melhor aproximar a distribuição *a posteriori* $p(\theta|y)$ (GELMAN et al., 2014).

Este método opera amostrando seqüencialmente valores de parâmetros de uma cadeia de Markov cuja distribuição estacionária é exatamente a distribuição *a posteriori* de interesse (CARLIN; LOUIS, 2009).

A amostragem é feita seqüencialmente, em que a distribuição das amostras obtidas depende do último valor extraído; sendo assim, as extrações formam uma cadeia de Markov. Trata-se de uma sequência de variáveis aleatórias θ^1, θ^2 . . . em que para qualquer t, a distribuição de θ^t, considerando todos os θ anteriores, depende apenas do valor mais recente, ou seja, θ^{t-1}. Então, inicia-se de um ponto θ^0 e para cada t obtem-se θ^t de uma distribuição de transição $T_t(\theta^t|\theta^{t-1})$, que é dependende de θ^{t-1}. As distribuições de transição devem ser construídas para que a cadeia de Markov convirja para uma distribuição estacionária única, que é a distribuição *a posteriori* $p(\theta|y)$. O sucesso do método ocorre pelo fato de que as distribuições aproximadas são melhoradas a cada passo da simulação, o que significa que vai convergindo para a distribuição real (GELMAN et al., 2014).

Os métodos designados Monte Carlo via Cadeias de Markov apoiam-se em amostras de θ dependentes, isto implica em resultados assintóticos mais complexos e maior número de iterações do que no método de Monte Carlo tradicional (PAULINO *et al.*, 2018).

Para a estimação dos parâmetros dos dados simulados via inferência Bayesiana foi utilizado o JAGS (*Just Another Gibbs Sampler*), que constitui de um programa para análise de modelos Bayesianos usando MCMC. É escrito em C++, porém, possui interface com o *software* R pelo pacote *rjags*, o qual foi utilizado para

esta aplicação. O JAGS gera a amostra a partir da distribuição *a posteriori* dada a amostra observada. O número de cadeias pode ser definido no código e produz uma sequência independente de amostras a partir das distribuições *a posteriori* selecionadas para cada parâmetro do modelo. A saída do MCMC é dividida em duas partes, a primeira é o "*burn in",* este período pode ser selecionado para descarte, é o intervalo entre o valor inicial e o início do período a ser considerado, ou seja, um período adaptativo. A segunda parte ocorre desde este ponto até o final das iterações, em que é considerado que a saída convergiu (PLUMMER, 2017).

Os métodos básicos de simulação de cadeia de Markov são o *Gibbs sampler* e o *Metropolis-Hastings*. O algoritmo *Gibbs sampler* também denominado amostragem condicional completa, gera uma sequência de amostras da distribuição conjunta de probabilidades inicial $\theta_1, \theta_2 \dots \theta_k$. Em cada iteração dos ciclos Gibbs é extraído um subconjunto condicional a todos os outros amostrados até o momento, por meio do subvetor inicial de θ (GELMAN et al., 2014).

Para os dados simulados da rede neural *Multilayer Perceptron* foram consideradas possibilidades para 1 a 5 neurônios na camada oculta. Sendo que para um neurônio, por exemplo, tem-se as equações (27) e (28), referentes as observações amostrais. Na equação (29), tem-se *a priori* dos parâmetros sendo uma distribuição normal de média 0 e variância 10, para que a busca seja maior.

$$\mu_i = \quad w_{02} + w_{12}\frac{1}{(1+e^{-(w_{01}+w_{11}x_{i1}+w_{21}x_{i2}+w_{31}x_{i3}+w_{41}x_{i4}+w_{51}x_{i5})})} \tag{27}$$

$$y_i|\underline{w} \sim N(\mu_i, 1) \tag{28}$$

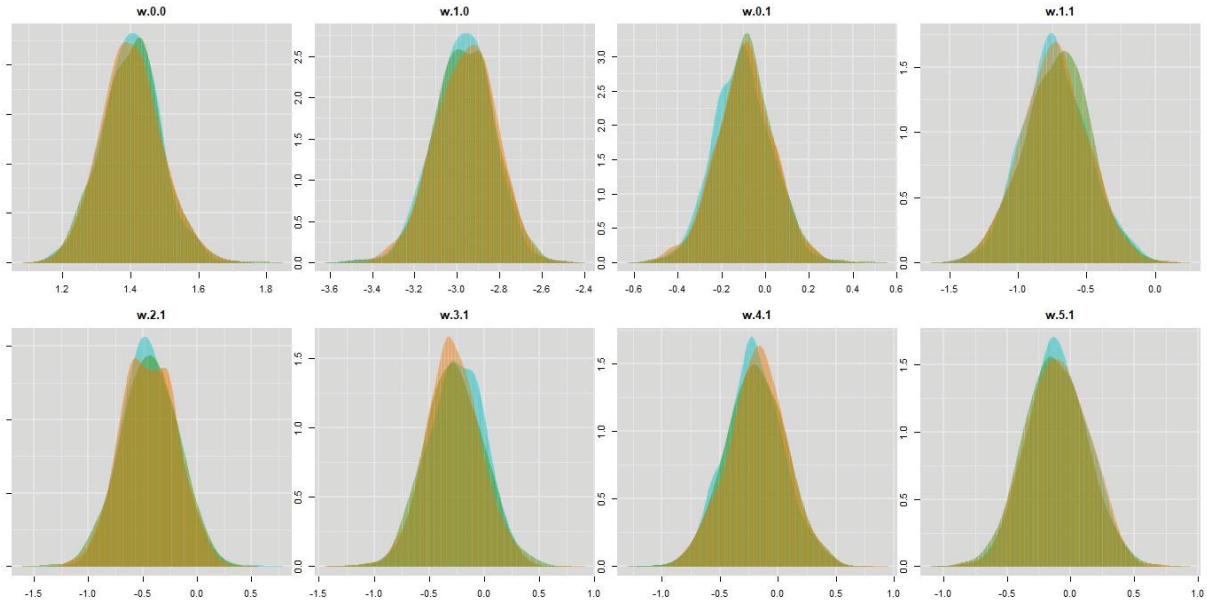
$$w_{hj} \sim N(0{,}10) \quad h = 0, \dots, 5 \quad j = 1{,}2 \tag{29}$$

A lógica é a mesma para as tentativas com mais neurônios na camada oculta. Com os dados simulados, foi observado o treinamento no contexto de inferência Bayesiana a fim de compreender a rede neural. Sabendo que a média dos dados foi gerada de um modelo MLP com 2 neurônios na camada oculta e cuja função de ativação é sigmóide logística, foi avaliado o treinamento com esta configuração e também de estruturas que não é a da geração dos dados, ou seja, com 1, 3, 4 e 5 neurônios na camada oculta.

Com 1 neurônio na camada oculta os resultados da distribuição *a posteriori* podem ser observados na FIGURA 13. Tem-se então a distribuição encontrada para

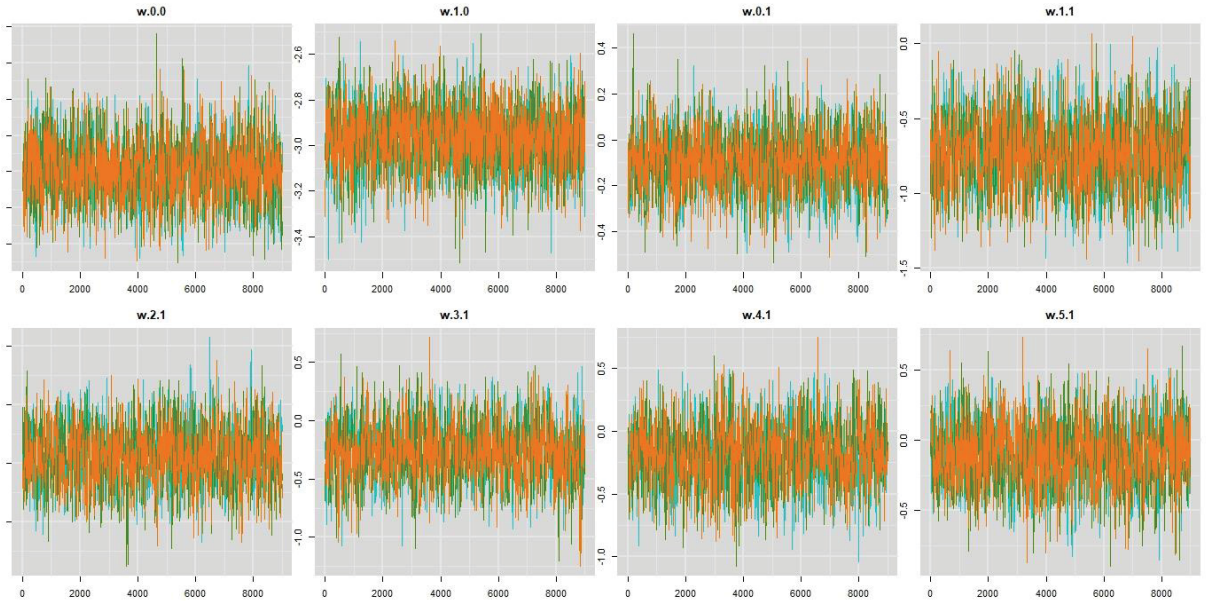
os parâmetros, neste caso as três cadeias obtiveram resultados semelhantes para valores dos parâmetros. As cadeias de Markov podem ser observados na FIGURA 14, também pode-se perceber um certo alinhamento entre as cadeias.

FIGURA 13 - DISTRIBUIÇÃO *A POSTERIORI* COM 1 NEURÔNIO



FONTE: A autora (2020).

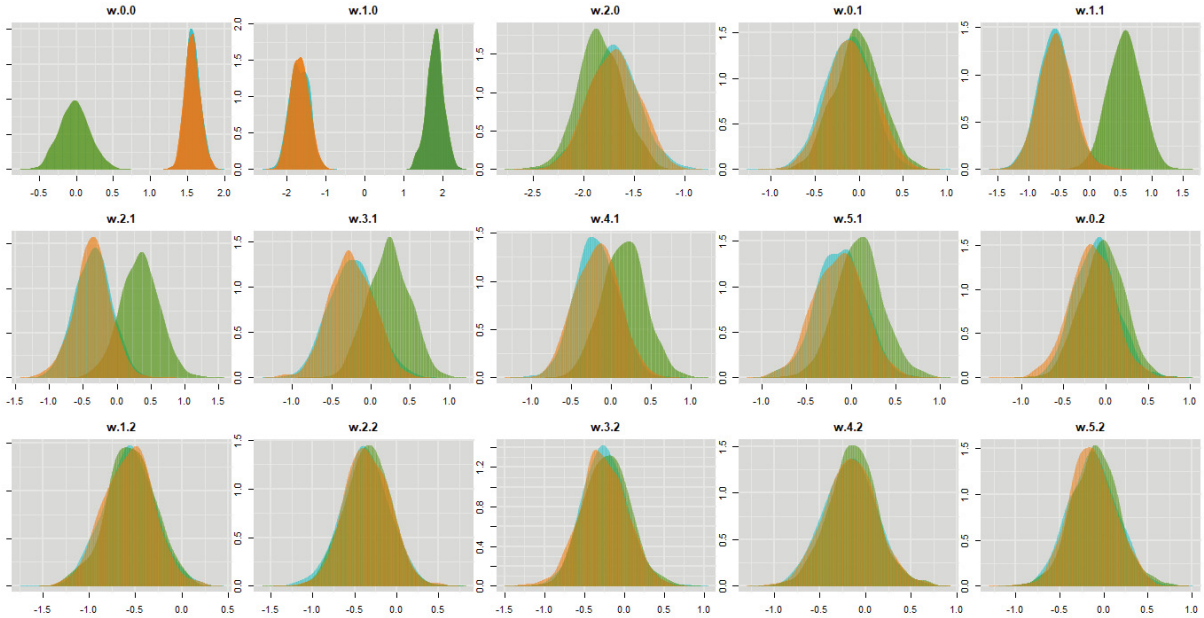
FIGURA 14 - CADEIAS DE MARKOV COM 1 NEURÔNIO



FONTE: A autora (2020).

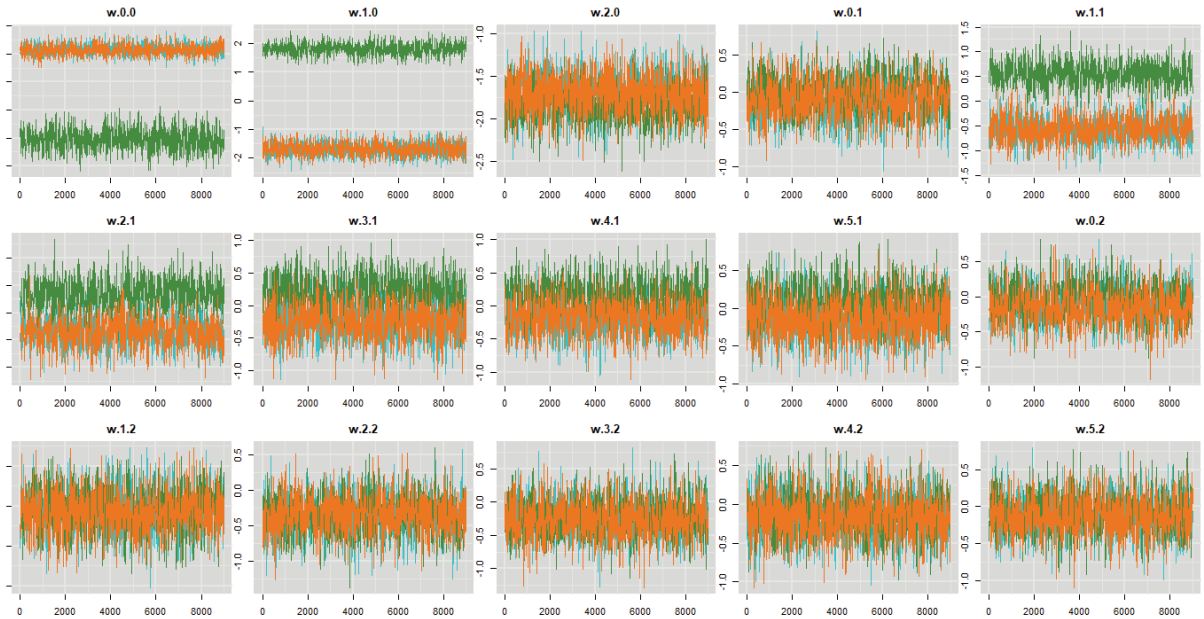
Posteriormente a mesma simulação foi aplicada considerando 2 neurônios na camada oculta, ou seja, com a configuração geradora dos dados, a fim de avaliar como seria a estimação dos parâmetros para este caso. Pode-se observar nas FIGURAS 15 e 16 a distribuição *a posteriori* e as cadeias de Markov, respectivamente.

FIGURA 15 - DISTRIBUIÇÃO *A POSTERIORI* COM 2 NEURÔNIOS



FONTE: A autora (2020).

FIGURA 16 - CADEIAS DE MARKOV COM 2 NEURÔNIOS

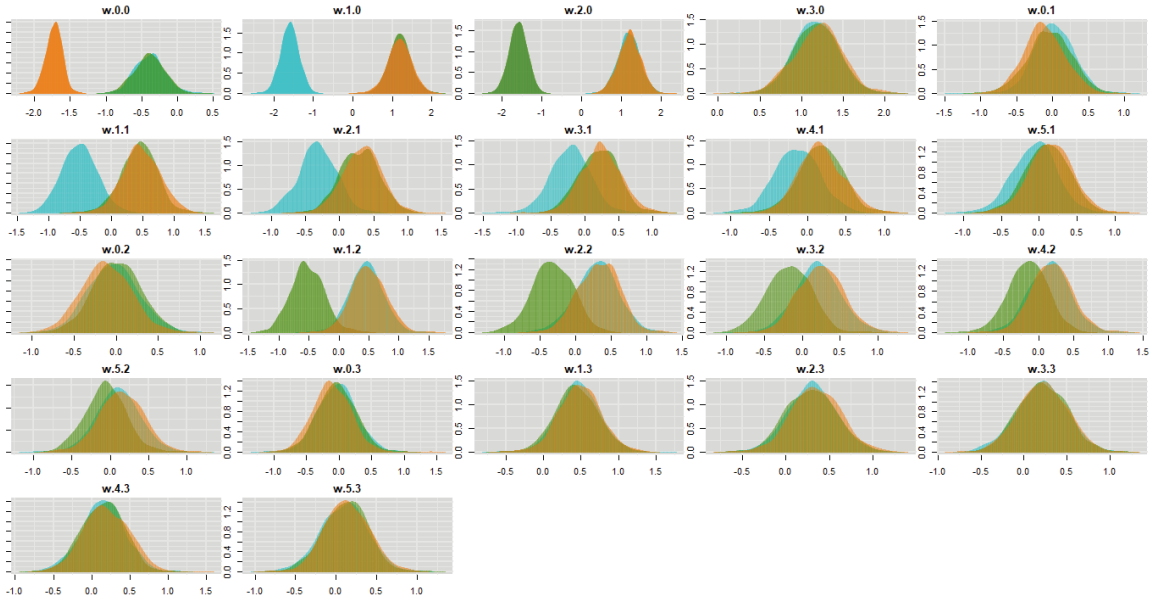


FONTE: A autora (2020).

É possível notar que o aprendizado nas cadeias foi pior nos parâmetros w_{00}, w_{01} e w_{02}, os quais correspondem a conexão dos neurônios ocultos com a camada de saída e o intercepto. Nos demais parâmetros ainda assim as cadeias também não foram completamente estáveis.

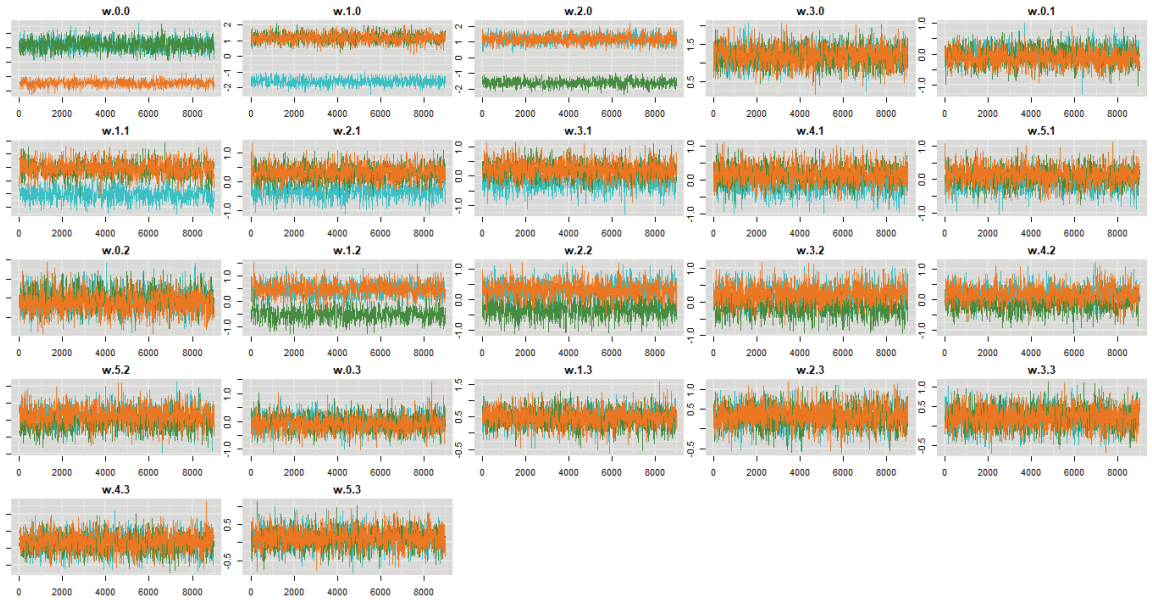
Foi aplicada também inferência Bayesiana com 3 neurônios na camada oculta, que pode ser observada nas FIGURAS 17 e 18.

FIGURA 17 - DISTRIBUIÇÃO *A POSTERIORI* COM 3 NEURÔNIOS



FONTE: A autora (2020).

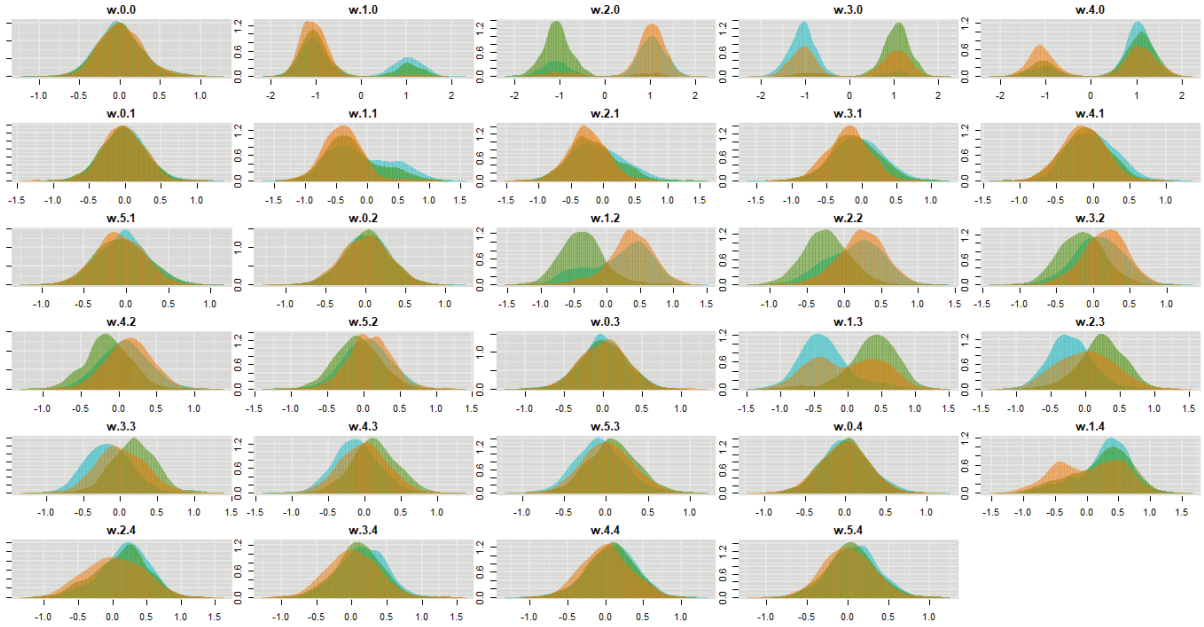
FIGURA 18 - CADEIAS DE MARKOV COM 3 NEURÔNIOS



FONTE: A autora (2020).

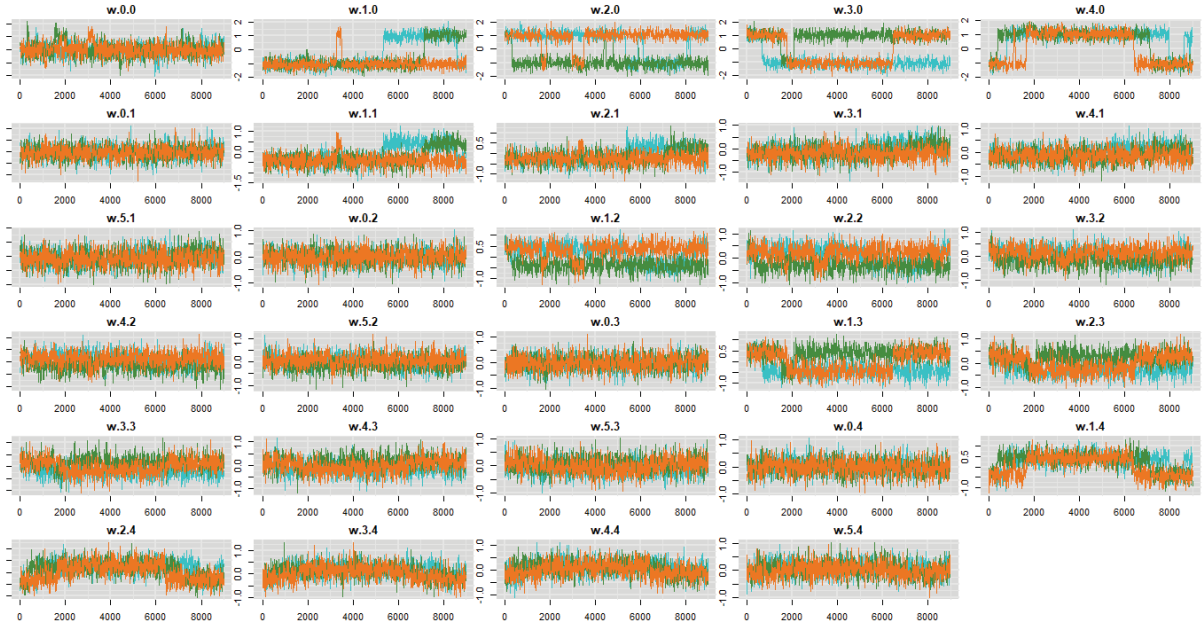
A aplicação com 4 neurônios na camada oculta, pode ser observada nas FIGURAS 19 e 20.

FIGURA 19 - DISTRIBUIÇÃO *A POSTERIORI* COM 4 NEURÔNIOS



FONTE: A autora (2020).

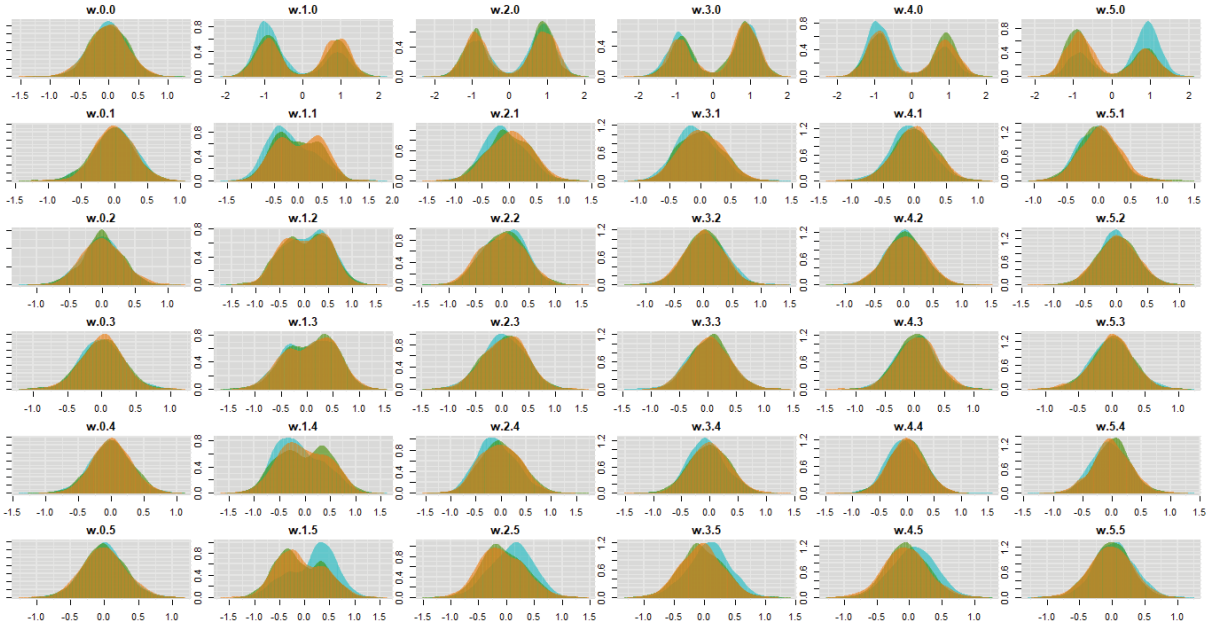
FIGURA 20 - CADEIAS DE MARKOV COM 4 NEURÔNIOS



FONTE: A autora (2020).

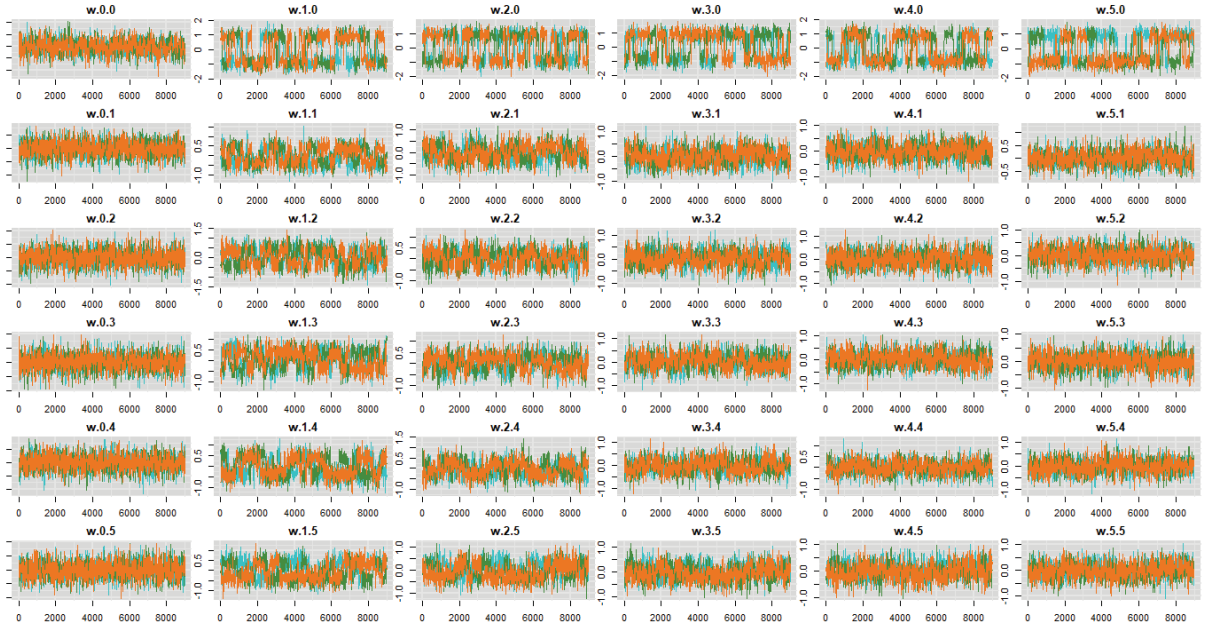
Foi aplicada também inferência bayesiana com 5 neurônios na camada oculta, conforme as FIGURAS 21 e 22. Neste caso é possível verificar inclusive distribuições bimodais nas cadeias para determinados parâmetros, o que evidencia a dificuldade de encontrar um único valor ótimo.

FIGURA 21 - DISTRIBUIÇÃO *A POSTERIORI* COM 5 NEURÔNIOS



FONTE: A autora (2020).

FIGURA 22 - CADEIAS DE MARKOV COM 5 NEURÔNIOS



FONTE: A autora (2020).

Com as informações obtidas *a posteriori* é possível fazer não somente uma previsão pontual, mas construir um intervalo de credibilidade para a previsão. Assim como há os intervalos de confiança em estatística freqüentista, no contexto bayesiano tem-se intervalos de probabilidade *a posteriori*, denominados de credibilidade. Em geral, os intervalos de credibilidade bayesianos não coincidem com os intervalos de confiança freqüentistas dado que o primeiro incorpora a informação contextual específica do problema da distribuição *a priori*. A região de credibilidade é definida numericamente, não sendo aleatória e admite uma interpretação probabilística (CARLIN; LOUIS, 2009; PAULINO *et al*., 2018).

A sessão 2.5 constitui de uma breve descrição a respeito do algoritmo *simulated annealing*, o qual faz parte de uma etapa do modelo proposto, o qual está apresentado na seção 2.7.

2.3.5 SIMULATED ANNEALING

Simulated Annealing (SA) é um algoritmo estocástico de busca local que, a partir de alguma solução inicial, explora iterativamente a vizinhança da solução atual. Trata de um método de otimização inspirado no trabalho de Metropolis *et al*. (1953), que propuseram uma integração de Monte Carlo para resolver problemas de estatística mecânica, esta consiste na disciplina que estuda a física da matéria condensada por meio de métodos para avaliar átomos encontrados em matéria líquida ou sólida. A ideia é submeter certos materiais inicialmente a altas temperaturas e reduzir gradualmente até atingirem, com aumentos e reduções do estado de energia, o equilíbrio térmico. Então, é possível melhorar soluções e piorar probabilisticamente de acordo com a quantidade de deterioração e de um parâmetro denominado temperatura, a qual diminui com o tempo, assim como a probabilidade de uma partícula se mover. Isto ocorre até que o sistema atinja um estado de energia mais baixa, que consiste em seu estado fundamental (KIRKPATRICK *et al*., 1983; FRANZIN; STÜTZLE, 2019).

Kirkpatrick *et al*. (1983) e Aragon *et al*. (1984) transformaram essas idéias em um método heurístico para lidar com problemas de otimização combinatória. Para cada configuração tem-se um conjunto de posição atômica $\{r_i\}$ e também um fator de probabilidade de Boltzmann, dado por $\exp\left(-\frac{E\{r_i\}}{k_B T}\right)$, sendo $E\{r_i\}$ a energia da

configuração, k_B a constante de Boltzmann e T a temperatura. A cada etapa do algoritmo, tem-se uma nova configuração e a mudança de energia é computada ΔE, o que no contexto do método heurístico corresponde à função custo. Caso haja uma redução na energia, a nova configuração é aceita, do contrario é feita uma verificação de forma probabilística. Tem-se então a $P(\Delta E) = \exp\left(-\frac{\Delta E}{k_B T}\right)$ e com isto o sistema envolve a distribuição Boltzmann. Uma forma de implementar a parte aleatória é com a seleção de um número de uma distribuição uniforme com intervalo (0,1). Se o número sorteado for menor do que $P(\Delta E)$, então a nova configuração segue para o próximo passo, caso contrário segue a anterior (KIRKPATRICK *et al*., 1983; ARAGON *et al*, 1984).

Para o cálculo da temperatura há diferentes métodos. Na função *optim* do pacote *stats* a temperatura diminui de acordo com o esquema de arrefecimento logarítmico, conforme um resfriamento algoritmo indicado por Belisle, 1992. Sendo assim, a temperatura é ajustada para $\varphi/\log\left(\left(\frac{(t-1)}{\alpha}\right) * \alpha + \exp(1)\right)$, em que t corresponde a iteração atual e φ e α são controlados e, por padrão, o valor corresponde a 10 (BELISLE, 1992)

2.3.6 SCORE INFORMATION CRITERIA

O cálculo de uma rede neural, no geral, consiste da multiplicação de variáveis da camada de entrada por diversos parâmetros, também denominados pesos. Todas as variáveis devidamente ponderadas passam por todos os neurônios que estão na camada oculta, que por fim também são ponderados por parâmetros em direção a camada de saída, além destes parâmetros mencionados há os interceptos, denominados *bias*.

Considerando o modelo de rede neural *Multilayer Perceptron*, estudada no presente trabalho, sendo p o número de variáveis na camada de entrada e z o número de neurônios na camada oculta, com apenas uma camada neste exemplo, a quantidade de parâmetros pode ser calculada conforme $(p+1)z + z + 1$, sendo assim, na situação com 5 variáveis e 2 neurônios na camada oculta tem-se 15 parâmetros para serem estimados.

Neste contexto de modelos superparametrizados, é abordado o método *Score Information Criteria* (SIC), proposto por Stoklosa *et al*. (2014). O método pode ser

usado quando o modelo constitui de um número grande de variáveis para a composição do preditor linear e a principal vantagem está no fato de que pode ser calculado para todos os modelos candidatos sem a necessidade de ajustá-los (BONAT *et al.*, 2017; STOKLOSA *et al.*, 2014).

Este critério é uma função da estatística escore U, definida por (30) e cujas proporiedades é dada em (31), sendo que $D_i^T = \frac{\partial \mu_i}{\partial w}$ e $V_i^{-1} = \frac{1}{\sigma^2}\mathrm{I}$. A distribuição da função escore U é assintóticamente gaussiana multivariada de média zero e matriz de variância J, que é a informação de Fisher (STOKLOSA *et al.*, 2014).

$$U(w) = \sum_{i=1}^{N} D_i^T V_i^{-1}\big(y_i - \mu_i(w)\big), \tag{30}$$

$$E\big(U(w)\big) = 0, \quad \mathrm{E}\left(-\frac{\partial U}{\partial \beta}\right) = \sum_{i=1}^{N} D_i^T V_i^{-1} D_i = J. \tag{31}$$

Supondo que o vetor de parâmetros θ pode ser particionado como $w = (w_1^T, w_2^T)^T$, cujas dimensões são $(p-r)\times 1$ e $r \times 1$ respectivamente. Então, a função escore U e sua matriz de informação J podem também ser particionadas para (32) e (33) (NUAMAH *et al.*, 1996)

$$U(w) = (U_1^T(w), U_2^T(w))^T, \tag{32}$$

$$J = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}. \tag{33}$$

A hipótese nula é dada por $H_0: w_2 = 0$ e, sendo $\overline{w} = (\overline{w}_1^T, 0)^T$ um vetor estimado sob H_0. O escore é então $\widehat{U}(\widehat{w}) = (\widehat{U}_1^T(\widehat{w}), \widehat{U}_2^T(\widehat{w}))^T$ = $(0^T, \widehat{U}_2^T(\widehat{w}))^T$ em que $U_2(\widehat{w}) = \sum_i \frac{\partial \mu_i^T}{\partial w_2} \hat{V}_i^{-1}(y_i - \hat{\mu}_i)$ então, o teste escore generalizado para H_0 é dado por (34) sendo que $\hat{V}_{U_2}$é a matriz de variância do subvetor $\widehat{U}_2(\widehat{w})$ e pode ser calculada pelas submatrizes conforme a equação (35). T_U possui distribuição Qui-quadrado com r graus de liberdade (NUAMAH *et al.*, 1996)

$$T_U = \widehat{U}_2^T(\widehat{w}) \hat{V}_{U_2}^{-1} \widehat{U}_2(\widehat{w}), \tag{34}$$

$$\hat{V}_{U_2} = \hat{J}_{22} - \hat{J}_{21}\hat{J}_{11}^{-1}\hat{J}_{12} \tag{35}$$

Stoklosa *et al*. (2014) fizeram uso da estatística escore generalizada para aproximar um critério baseado na verossimilhança. O critério de informação baseado na verossimilhança, por Burnham e Anderson (1999) é dado em (36), sendo $\lambda > 0$ para alguma função de log-verossimilhança definida $l(\theta; y)$, como neste caso a $l(\theta; y)$ não é definida então a equação é representada pela estatística da razão da log-verossimilhança com um termo de penalidade (37). Então, a principal ideia do método é substituir o primeiro termo pela estatística escore, motivado pela aproximação quadrática da estatística de razão de verossimilhança, tem-se então o SIC de um passo (38) (BURNHAM, ANDERSON, 1999; STOKLOSA *et al.*, 2014).

$$IC(w; y) = -2l(w; y) + \lambda|w| \tag{36}$$

$$-2l(w; y) + \lambda|w| = -2\{l(w; y) - l(\hat{w}_1; y)\} + \lambda|w| + c \tag{37}$$

$$SIC^{(1)}(w; y) = -T_U + \lambda|w| \tag{38}$$

Para o contexto das RNAs tem-se que $\underline{y}|X \sim N(\underline{\mu}, \Sigma)$ sendo $\underline{y}$ e $\underline{\mu}$ cada qual um vetor $n \times 1$, X uma matriz $n \times p$ e Σ uma matriz $n \times n$. A função log verossimilhança é dada pela equação (39). O vetor $\underline{\mu}$ constitui de n equações de μ_i (40), este é resultado da multiplicação da matriz $n \times (k+1)$, $k = 1, \ldots m$, denominada $\tilde{X}$, por um vetor de dimensão $(k+1) \times 1$ composto pelas saídas dos k neurônios da camada oculta e o termo *bias*.

$$l\left(\underline{\mu}, \Sigma, \underline{y}\right) = -\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma| - \frac{1}{2}(\underline{y} - \underline{\mu})\Sigma^{-1}(\underline{y} - \underline{\mu})^T \tag{39}$$

$$\underline{\mu} = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_n \end{bmatrix} = \begin{bmatrix} 1 & \frac{1}{1+e^{-(\underline{X}_1 w_{.1})}} & \cdots & \frac{1}{1+e^{-(\underline{X}_1 w_{.m})}} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \frac{1}{1+e^{-(\underline{X}_n w_{.1})}} & \cdots & \frac{1}{1+e^{-(\underline{X}_n w_{.m})}} \end{bmatrix} \begin{bmatrix} w_{00} \\ w_{10} \\ \vdots \\ w_{m0} \end{bmatrix} = \tilde{X} w_{k0} \tag{40}$$

As primeiras derivadas parciais para este contexto são obtidas conforme as equações (41) e (42) e o vetor $U_w = (U_{w_{.0}}, (U_{w_{.1}}, \ldots, U_{w_{.m}}))$ possui dimensão $1 \times ((p+1) * k + k + 1)$.

$$U_{w_{.0}} = \frac{\partial l(\mu)}{\partial \mu}\frac{\partial \mu}{\partial w_{.0}} = \frac{1}{\sigma^2}\mathrm{I}(\underline{y} - \tilde{X}w_{.0})^T\tilde{X} \tag{41}$$

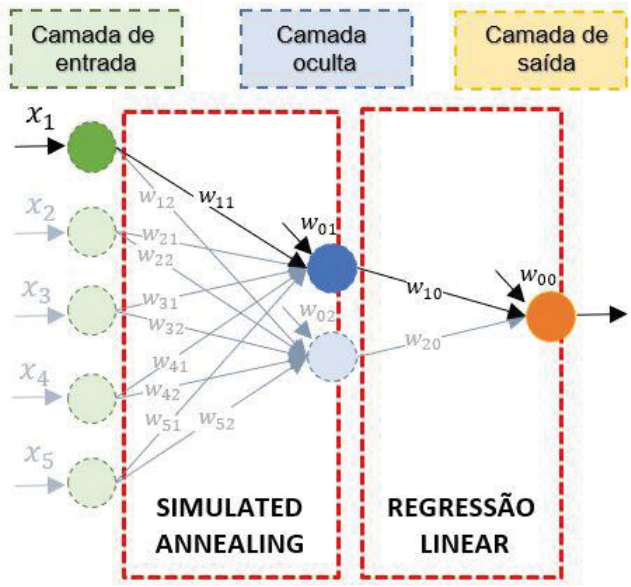
$$(U_{w_{.1}}, \dots, U_{w_{.m}}) = U_{w_{.k}} = \frac{\partial l(\mu)}{\partial \mu}\frac{\partial \mu}{\partial \tilde{X}}\frac{\partial \tilde{X}}{\partial w_{.k}} = \frac{1}{\sigma^2}\mathrm{I}\left(\underline{y} - \tilde{X}w_{.0}\right)^T w_{.0}\frac{X^T(e^{-Xw_{.k}})^T}{(1+e^{-Xw_{.k}})^2} \tag{42}$$

O algoritmo proposto para o contexto das redes neurais consiste na aplicação do SIC em etapas, avaliando as covariáveis que entram em cada neurônio, uma a uma.

Inicialmente é verificada qual covariável possui maior correlação com a variável resposta e esta inicia o modelo juntamente com o termo *bias* constituindo uma primeira configuração de rede com um neurônio. Após isto, os parâmetros da camada de entrada para a oculta, que nesta configuração inicial correspondem ao *bias* e a covariável de maior correlação, são estimados pelo algoritmo *simulated annealing* (SA). Os demais parâmetros – correspondentes à camada de saída – são estimados pela regressão linear, sendo atualizados dentro do algoritmo SA.

Para melhor compreensão, a FIGURA 23 apresenta a configuração da rede habilitada inicialmente, supondo, para esta ilustração, que x_1 seja a covariável de maior correlação com a resposta.

FIGURA 23 - CONFIGURAÇÃO INICIAL DO MODELO POPOSTO



FONTE: A autora (2020).

Após a estimação é verificado se alguma variável entra no modelo, nesta etapa faz-se o *ranking* SIC com todas as variáveis restantes. Aquela de valor de SIC mais negativo entra no modelo e então todos os parâmetros são reestimados da mesma forma explicada anteriormente.

Dada a nova estrutura habilitada, são avaliadas as covariáveis que ainda não entraram no primeiro neurônio, então, faz-se novamente o *ranking* SIC e a reestimação e assim ocorre até que não haja mais valores negativos no *ranking,* sempre reestimando todos os parâmetros quando uma nova covariável passa a fazer parte do modelo.

Após finalizar o processo no primeiro neurônio, inicia-se a configuração no segundo da mesma forma que iniciada no primeiro e também as estimações ocorrem igualmente e são feitas em todos os parâmetros existentes no modelo.

Caso ocorra de já no primeiro *ranking* SIC não haver nenhum valor negativo, então não exclui-se imediatamente a covariável que iniciou a configuração daquele neurônio, mas sim, é avaliado se o erro desta estrutura, que calculado conforme descrito anteriormente, foi inferior ao da anterior, se for inferior, permanece o neurônio habilitado com a estrutura inicial e avalia-se o próximo, caso contrário é excluída esta nova configuração e permane a anterior como modelo final, tem-se no pseudocódigo da FIGURA 24 a regra do algoritmo proposto.

FIGURA 24 – PSEUDOCÓDIGO DO ALGORITMO PROPOSTO

```
'A regressão linear está imbutida no modelo
Algoritmo Proposto
Início
Entradas: x_1, ..., x_k, y
   Para k = 1 até p faça
      cor(k) ← cor(x_k, y)
   Fim - Para
   x_b = max(cor(k))
Estima ← otimiza(w_0j, w_bj)*{algoritmo SA}
Para Neurônio <= j faça
      Enquanto min(ranking) < 0 e houver variável faça
         Para i = {demais variáveis} faça
               ranking [ , i] ← SIC(i){cálculo com os parâmetros estimados}
         Fim - Para
         Entra ← min(ranking)
         Estima ← otimiza(Estima, Entra)*{algoritmo SA}
      Fim - Enquanto
         Estima ← otimiza(Estima, Entra, w_0j, w_bj)*{algoritmo SA}
         Se erro j+1 < j Então
               Neurônio = Neurônio+1
         Senão
         Neurônio = j+1
Fim - Para
Fim
```

FONTE: A autora (2020).

Com os dados simulados foram então avaliadas as correlações entre a variável resposta e as 5 explicativas, conforme mostra a TABELA 3.

TABELA 3 - CORRELAÇÃO DAS COVARIÁVEIS COM A VARIÁVEL RESPOSTA

x_1	x_2	x_3	x_4	x_5
0.995	0.990	0.984	0.979	0.973

FONTE: A autora (2020).

Como x_1 possui maior correlação, inicia no primeiro neurônio. Os parâmetros que relacionam as covariáveis são estimados pelo algoritmo SA e os parâmetros da camada de saída pela regressão linear, tem-se então um modelo inicial conforme (43)

$$\mu_i = \quad 0.11 - 0.28\frac{1}{(1+e^{-(7.48-14.28x_{i1})})}. \tag{43}$$

As covariáveis x_2, x_3, x_4 e x_5, são testadas, cujos valores de SIC estão na TABELA 4, referentes ao primeiro *ranking* do primeiro neurônio. A covariável x_2 foi a que obteve valor mais negativo e portanto está agora no modelo, são então estimados os parâmetros do novo modelo. O próximo passo então é testar se x_3, x_4 e x_5 também entram no primeiro neurônio. Como o valor do segundo *ranking* SIC para todas as covariáveis foi positivo, então nenhuma outra covariável entra.

TABELA 4 - *RANKING SIC* PRIMEIRO MODELO

	1º Neurônio		**2º Neurônio**	**3º Neurônio**
	1^oSIC	2^oSIC	1^oSIC	1^oSIC
x_2	**-10.03**	-	9.93	11.94
x_3	-0.50	7.93	11.29	15.13
x_4	-4.42	7.12	11.91	15.63
x_5	5.52	6.44	4.85	15.77

FONTE: A autora (2020).

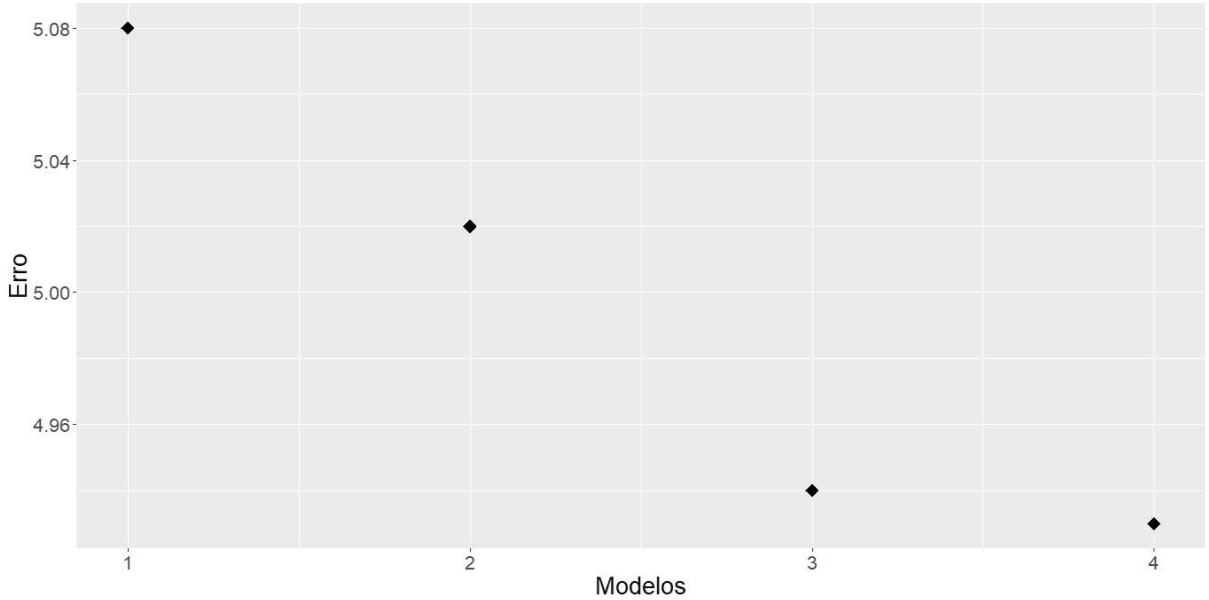
A avaliação então é iniciada no segundo neurônio, o qual tem o intercepto e a covariável $x_1$como informações iniciais. Então todos os parâmetros são reestimados dada a nova estrutura. Faz-se o *ranking* para as demais variáveis no segundo neurônio e conforme SIC do segundo neurônio, dado na TABELA 5, não entrou nenhuma outra variável.

A fim de verificar se o segundo neurônio permanece com as variáveis estimadas faz-se o cálculo do erro, o qual foi menor que o obtido com o modelo anterior e por isso as variáveis iniciadas no segundo neurônio permanecem e a avaliação do terceiro neurônio é iniciada. O mesmo procedimento então é feito no neurônio três e, conforme pode-se verificar os valores de SIC, também não foram incluídas mais covariáveis. Como o erro deste novo modelo foi menor do que o do anterior, o neurônio 3 permanece e avalia-se o 4, o qual obteve erro maior e por isto o modelo final está definido, conforme a equação (44).

$$\mu_i = 4.99 + 0.45\frac{1}{(1+e^{-(-1.48-0.08x_{i1}+1.53x_{i2})})} - 12.24\frac{1}{(1+e^{-(-0.32-0.31x_{i1})})} + 0.05\frac{1}{(1+e^{-(-2.45-4.57x_{i1})})} \quad (44)$$

Os avanços até a construção do modelo final pelo algoritmo proposto podem ser verificados na FIGURA 25. Inicialmente com o primeiro neurônio somente com o intercepto e a variável de maior correlação, posteriormente com a inclusão da variável no primeiro neurônio, dado o resultado do *ranking*, depois com o 2 neurônio e após a inclusão do terceiro. O erro final do modelo foi de 4,93.

FIGURA 25 – ERROS DO MODELO SIC



FONTE: A autora (2020).

3 MATERIAL E MÉTODOS

Este capítulo é dedicado à descrição da metodologia empregada no estudo das redes neurais atificiais *Multilayer Perceptron* com três abordagens de aprendizado. A aplicação foi realizada em dois estudos de caso.

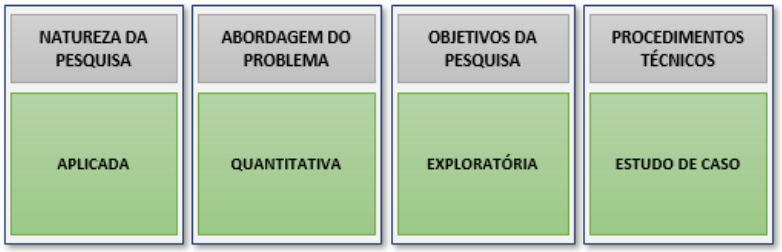
3.1 CLASSIFICAÇÃO DA PESQUISA

Segundo Gil (2008), pode-se definir pesquisa como o processo formal e sistemático de desenvolvimento do método científico. Sendo o objetivo fundamental da pesquisa descobrir respostas para problemas mediante o emprego de procedimentos científicos. Ainda conforme Gil (2002), toda e qualquer classificação se faz mediante à algum critério e, com relação às pesquisas, é usual a classificação com base em seus objetivos gerais.

Conforme Silva e Menezes (2005), há diversas formas de classificar a pesquisa, sendo que a apresentada pelo autor é a clássica, na qual foi classificado o presente trabalho e está ilustrado na FIGURA 26. Com relação à natureza, pode-se classificar em dois tipos: básica ou aplicada. O presente trabalho enquadra-se na natureza aplicada, a qual tem como objetivo gerar conhecimentos para aplicação prática e é dirigida à solução de problemas específicos. O presente trabalho trata de uma aplicação prática das técnicas de redes neurais artificiais para a previsão em duas bases de dados.

Com relação à forma de abordagem da pesquisa, pode ser qualitativa ou quantitativa. A quantitativa, considera que tudo pode ser quantificável e requer o uso de recursos e de técnicas estatísticas (SILVA; MENEZES, 2005), portanto os dados do estudo estão enquadrados nesta classificação.

FIGURA 26 - CLASSIFICAÇÃO DA PESQUISA



FONTE: A autora (2020).

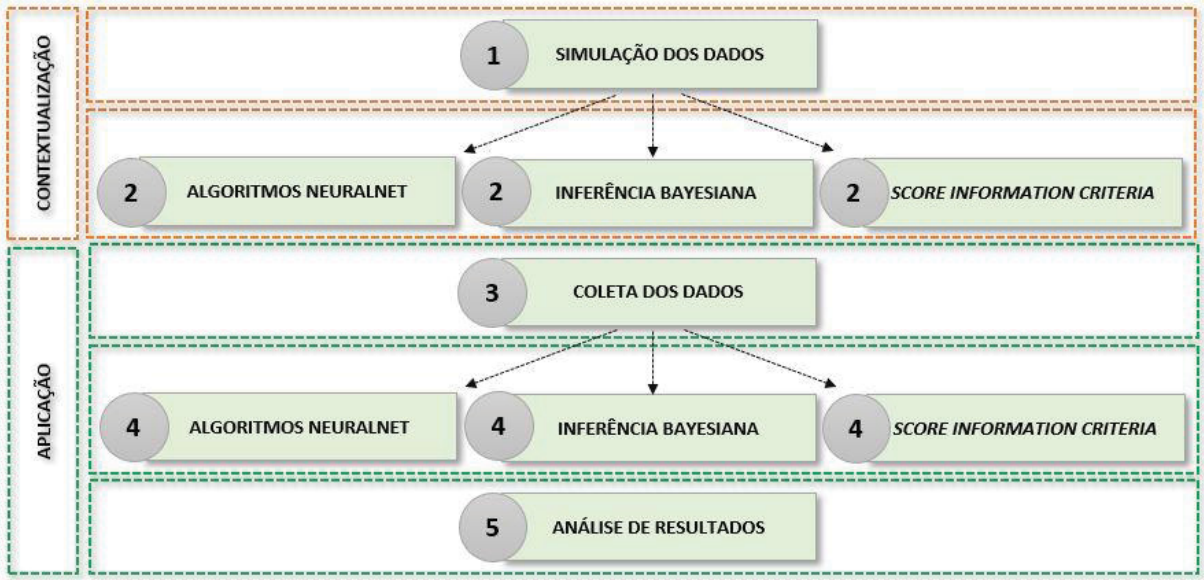
Quanto ao tipo de investição, é possível classificar as pesquisas em três grandes grupos, sendo eles: exploratórias, descritivas e explicativas (GIL, 2002). O presente trabalho é classificado como exploratório, estes assumem, em geral, as formas de pesquisas bibliográficas e estudos de caso, além de proporcionar maior familiaridade com o problema a fim de torná-lo explícito ou a construir hipóteses (SILVA; MENEZES, 2005).

Com relação aos procedimentos técnicos, podem ser classificados em: pesquisa bibliográfica, pesquisa documental, pesquisa experimental, levantamento, estudo de caso, pesquisa expost-fato, pesquisa-ação e pesquisa participante. Este trabalho pode ser classificado como estudo de caso, sendo que a aplicação foi em duas bases de dados.

3.2 ETAPAS DA PESQUISA

A pesquisa foi dividida em 5 etapas. As etapas 1 e 2 constituem de uma simulação de dados em uma rede neural MLP contextualizando a explicação das abordagens utilizadas para o aprendizado da rede e as demais etapas fazem parte da aplicação realizada em dois estudos de caso, conforme a FIGURA 27.

FIGURA 27 - ETAPAS DA PESQUISA

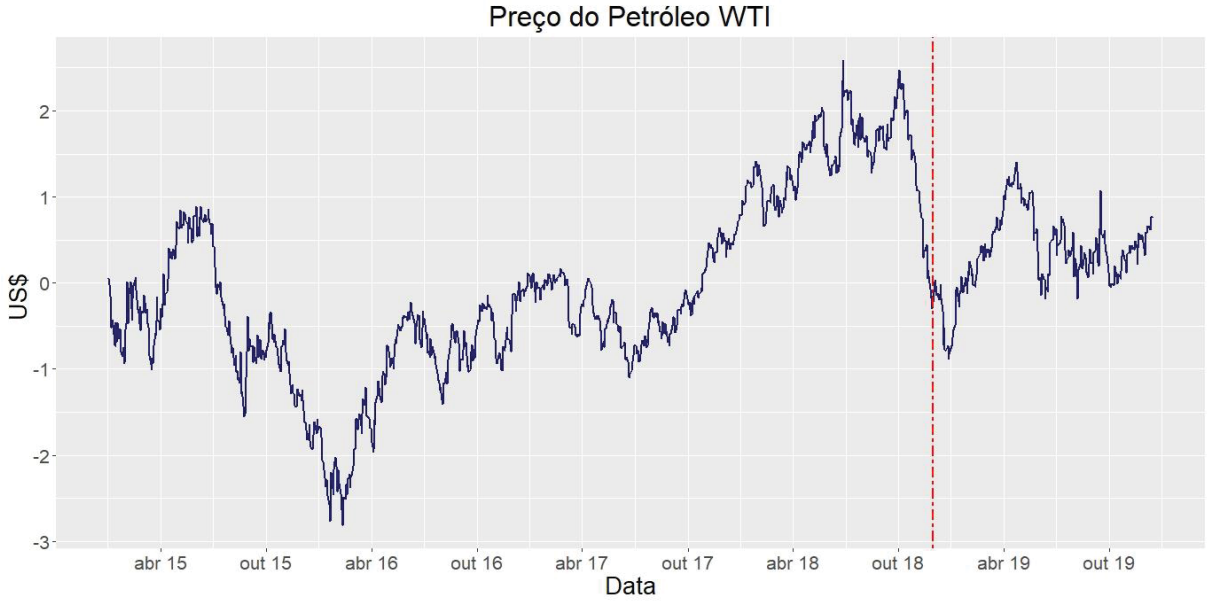


FONTE: A autora (2020).

Para a aplicação da metodologia foi escolhido o *software* R (R CORE TEAM, 2019), pois este é um *software* livre, o qual tem disponível diversas funções prontas, desde a função para coleta de dados da internet, até algumas implementações como o modelo de rede neural *Multilayer Perceptron* no pacote *neuralnet*.

As informações coletadas são referentes à duas aplicações. A primeira trata de dados históricos de preço diário de petróleo bruto WTI e corresponde ao período de 01 de janeiro de 2015 a 16 de dezembro de 2019, sendo utilizados para treinamento dos algoritmos 1.000 observações, aproximadamente 80%, e 262 para a validação. Para esta aplicação as covariáveis utilizadas foram os 5 dias anteriores ao de previsão, denotadas como t-1, t-2, t-3, t-4 e t-5, sendo que a primeira considera o valor do dia anterior, a segunda dois antes da previsão e assim ocorre até cinco dias anteriores ao da previsão. Na FIGURA 28 é possível verificar a série e o ponto de corte para treino e validação, correspondente a 30 de novembro de 2018.

FIGURA 28 - SÉRIE HISTÓRICA DE PETRÓLEO WTI

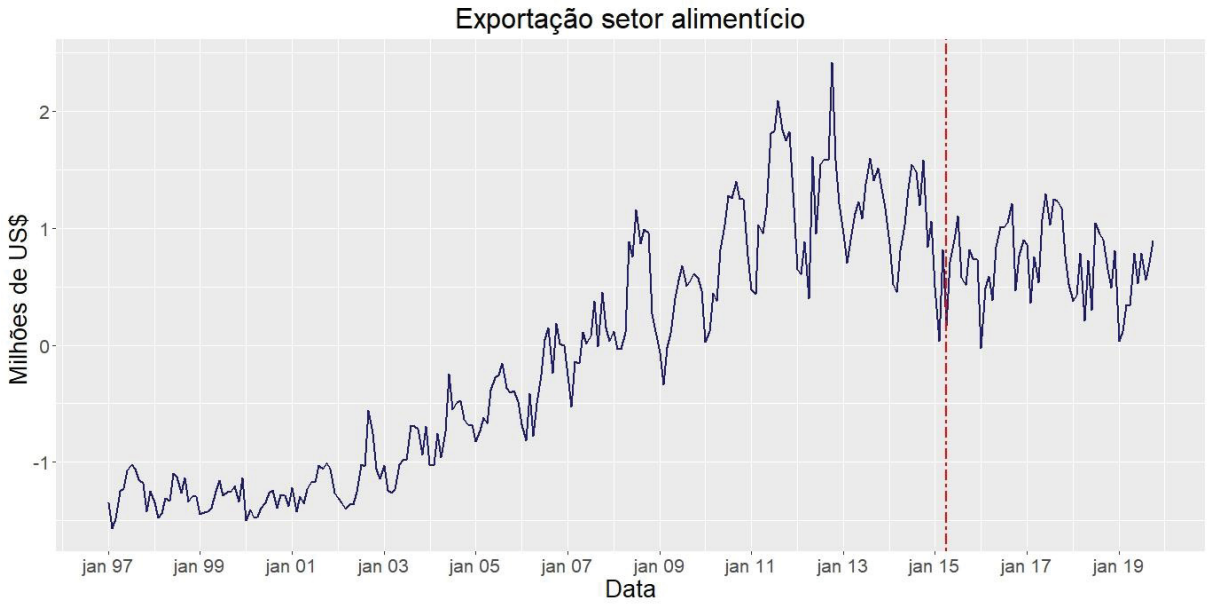


FONTE: A autora (2020)

A segunda aplicação é referente a previsão de exportação, em milhões de US$, do setor de produtos alimentícios no Brasil. As covariáveis utilizadas como explicativas constituem de dados econômicos: Índice Nacional de Preços ao Consumidor Amplo (IPCA) - alimentos e bebidas, Taxa de câmbio R$/US$ comercial, câmbio contratado em exportações, imposto sobre a importação e exportações. A

informação é mensal em todas as variáveis e o período considerado foi de janeiro de 1997 até outubro de 2019. Para treinamento foram utilizadas 80% das observações, ou seja, 220 meses e 54 para validação, o que cortou o período em abril de 2015, conforme pode ser observado na FIGURA 29.

FIGURA 29 - SÉRIE HISTÓRICA DE EXPORTAÇÃO DE ALIMENTOS



FONTE: A autora (2020).

Todas as informações utilizadas nas duas aplicações foram obtidas diretamente do pacote *ipeadatar* (GOMES, 2019) do *software* R e estão descriminadas no QUADRO 1.

QUADRO 1 - DETALHAMENTO DAS VARIÁVEIS NO *SOFTWARE* R

(continua)

Código	Descrição	Fonte
EIA366_PWTI366	Preço do pétroleo bruto WTI em US$.	[1]EIA
FUNCEX12_XVAL2N12	Exportações no setor produtos alimentícios (variável Exp_A): É excluido o pagamento de fretes, seguros, impostos e taxas de embarque, o dado está em milhões de US$.	[2]Funcex
PRECOS12_IPCAAB12	Índice Nacional de Preços ao Consumidor Amplo (variável IPCA) - alimentos e bebidas	[3]IBGE

QUADRO 1 - DETALHAMENTO DAS VARIÁVEIS NO *SOFTWARE* R

(conclusão)

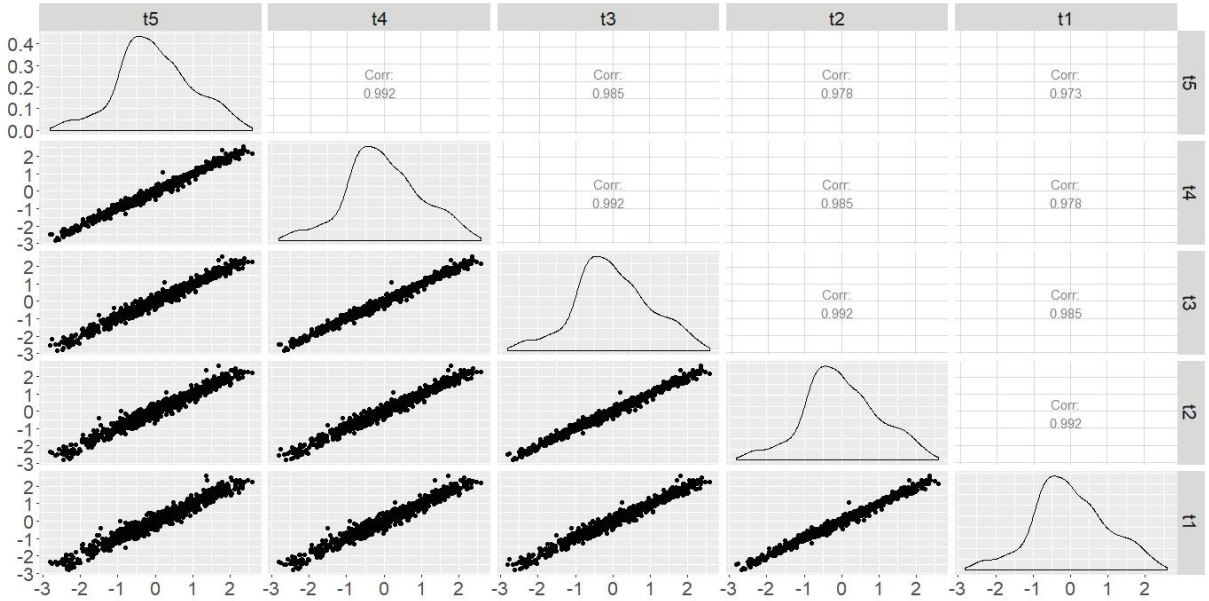
BM12_ERC12	Taxa de câmbio R$/US$ comercial (valor de compra) média do período (variável Taxa_C): São calculadas com base nas cotações diárias para a compra.	[4]Bacen
BM12_XCC12	Câmbio contratado – exportações (variável Câmbio_E):. Representa o somatório das contratações de compra de moeda estrangeira efetuadas pelos bancos comerciais junto ao mercado não financeiro relativas a exportações de bens.	[4]Bacen
SRF12_II12	Imposto sobre a importação(variável Imp): Receita bruta em milhões de US$.	[5]Min. Fazenda/SRF
SECEX12_XVTOT12	Exportação (variável Exp): Referente ao volume total do país excluindo pagamento de fretes, seguros, impostos e taxas de embarque, em milhões de US$.	[6]MDIC/SECEX

[1]Energy Information Administration (EIA) [2]Fundação Centro de Estudos do Comércio Exterior [3]Instituto Brasileiro de Geografia e Estatística [5]Ministério da Fazenda, Secretaria da Receita Federal [6] Ministério do Desenvolvimento, Indústria e Comércio Exterior-Secretaria de Comércio Exterior

FONTE: Elaboração própria com base nos dados do pacote ipeadatar.

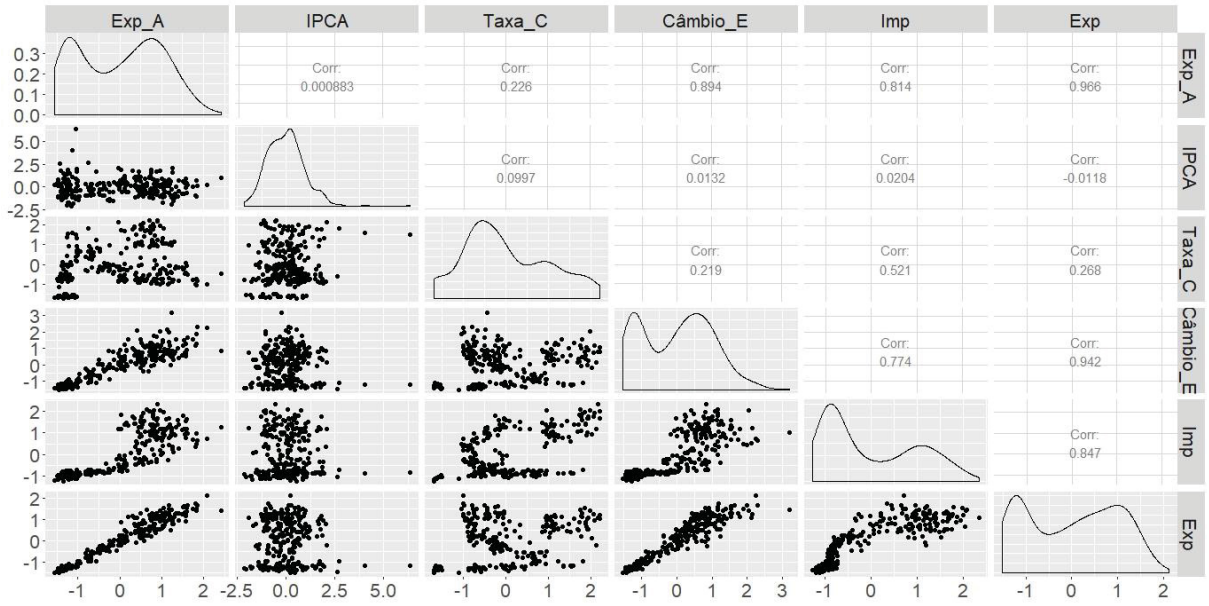
A correlação entre as variáveis nos estudos de caso 1 e 2 podem ser observadas nas FIGURAS 30 e 31.

FIGURA 30 - CORRELAÇÃO PETRÓLEO



FONTE: A autora (2020).

FIGURA 31 - CORRELAÇÃO EXPORTAÇÃO DE ALIMENTOS



FONTE: A autora (2020).

4 APRESENTAÇÃO DOS RESULTADOS

Esta seção está separada em dois estudos de caso. Inicialmente os dados apresentados correspondem a base de dados de preço de petróleo WTI e posteriormente a de exportação, em milhões de US$, no setor alimentício.

4.1 ESTUDO DE CASO 1: PREÇO DE PETRÓLEO WTI

Para o estudo de caso do preço de petróleo foram consideradas como covariáveis os 5 dias anteriores ao da previsão e tem-se 1.000 dias para treinamento. Foram avaliadas as arquiteturas de rede com 1 a 5 neurônios na camada oculta de uma rede MLP de função de ativação sigmóide logística.

Cada algoritmo treinou os dados 50 vezes. Na TABELA 5 tem-se os menores erros obtidos nas previsões geradas. Para o algoritmo *rprop+* o menor erro foi de 7,24, obtido com 36.166 iterações. Para o algoritmo *rprop-* com 62.303 iterações o menor erro foi de 7,06. O algoritmo *slr* somente convergiu as 50 previsões com 1 neurônio e deste o mínimo foi de 7,65, obtido com 4.849.032 iterações. O *sag* e *backprop* para esta aplicação também não convergiram as 50 vezes.

TABELA 5 – MENORES ERROS DOS ALGORITMOS NO ESTUDO DE CASO 1

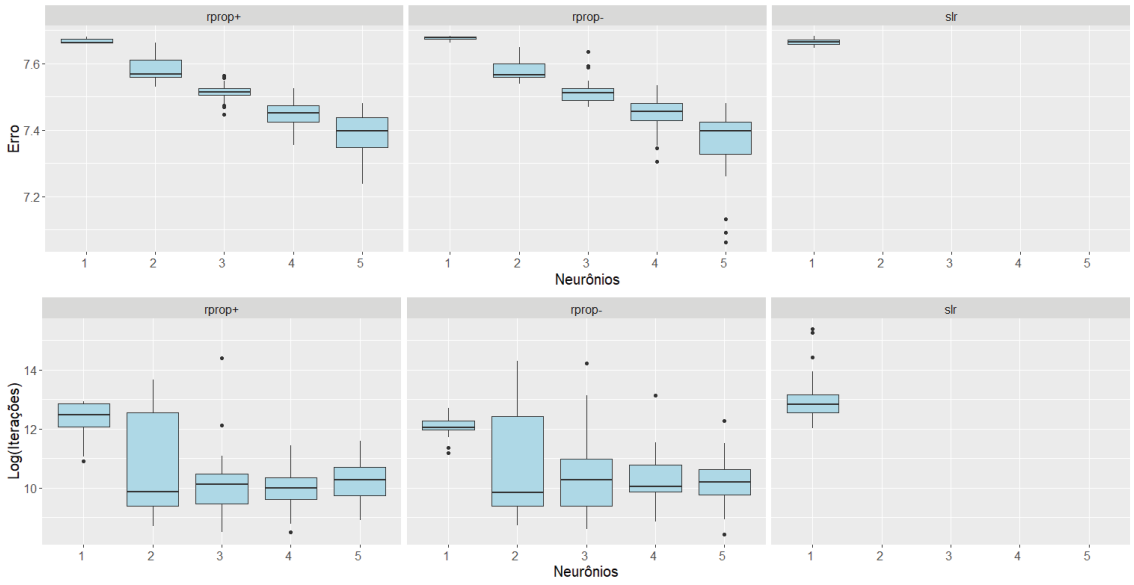
	rprop+	***rprop-***	***slr***
1 neurônio	7.66	7.66	**7.65**
2 neurônios	7.53	7.54	-
3 neurônios	7.45	7.47	-
4 neurônios	7.35	7.31	-
5 neurônios	**7.24**	**7.06**	-

FONTE: A autora (2020).

Com as 50 estimativas em cada algoritmo tem-se, na FIGURA 32, os erros e iterações, esta última dada em logaritmo. É possível verificar que para os algoritmos *rprop+* e *rprop-* o erro foi diminuindo conforme aumentou o número de neurônios, além disto os algoritmos que constituiam de 3 a 5 neurônios precisaram, no geral, de um menor número de iterações para convergirem. O *slr,* que somente obteve 50 convergências com um neurônio, obteve pior desempenho do que os demais

algoritmos, em volume de iterações e em erros se comparado a todas as configurações.

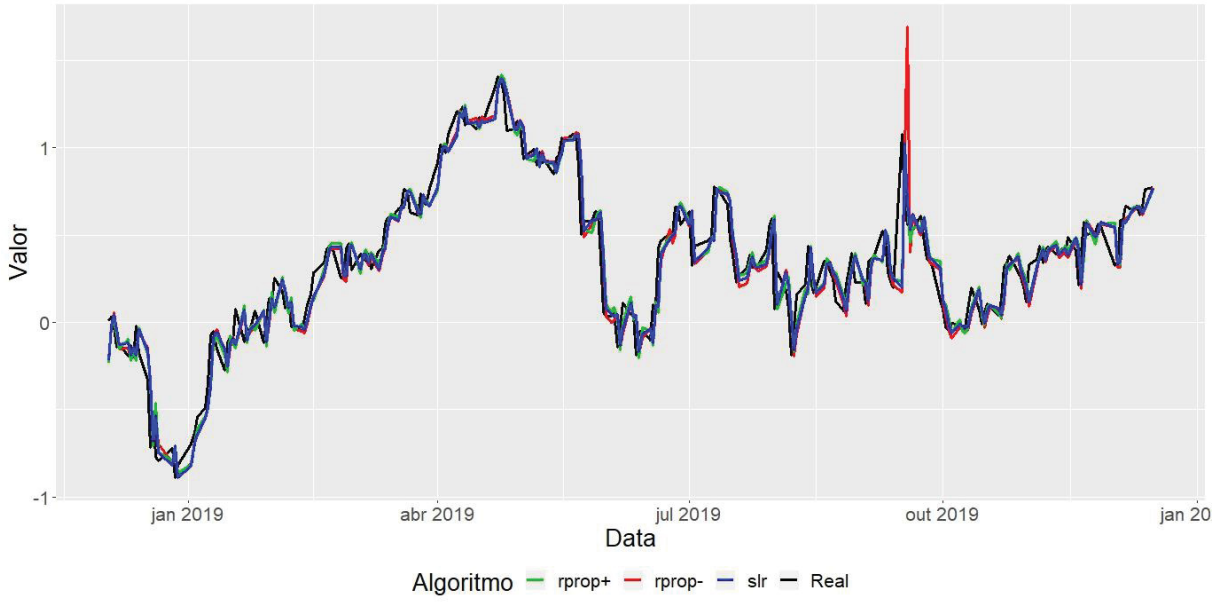
FIGURA 32 - BOXPLOT PARA ERRO E ITERAÇÕES



FONTE: A autora (2020).

Os parâmetros que resultaram em menor erro nos algoritmos foram aplicados para previsão nos dados de validação e pode-se observar na FIGURA 33 que não houve grandes discrepâncias nas previsões, exceto no algoritmo *rprop-*.

FIGURA 33 - PREVISÃO ALGORITMOS RNA



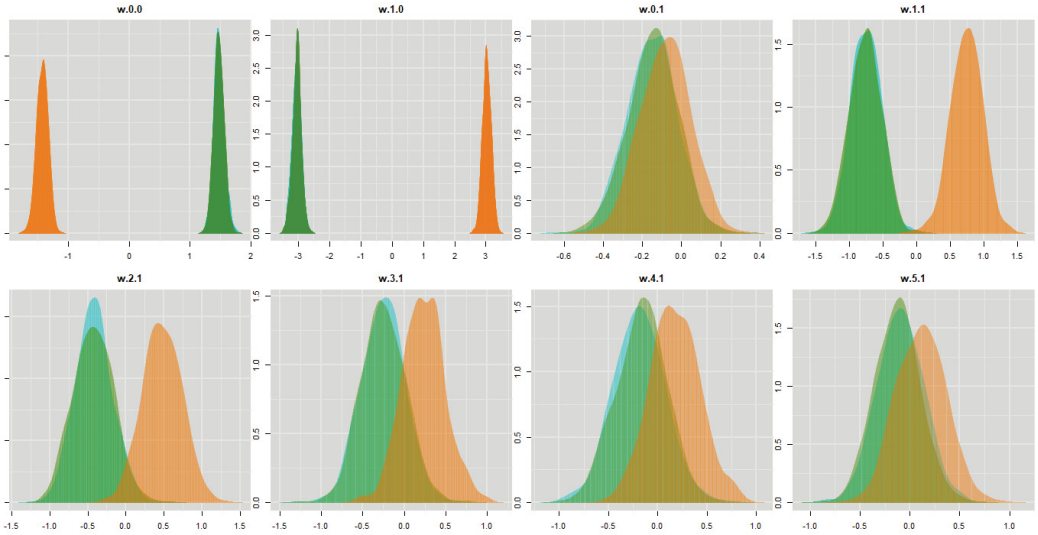
FONTE: A autora (2020).

Os erros obtidos na validação para *rprop+, rprop-* e *slr* foram de 2,39, 3,02 e 2,30, respectivamente e portanto, apesar de *slr* ter obtido maior erro no treinamento conseguiu prever melhor do que os demais.

Os parâmetros também foram estimados via Inferência Bayesiana com as tentativas de 1 a 5 neurônios na camada oculta de uma MLP cuja função de ativação é sigmóide logística.

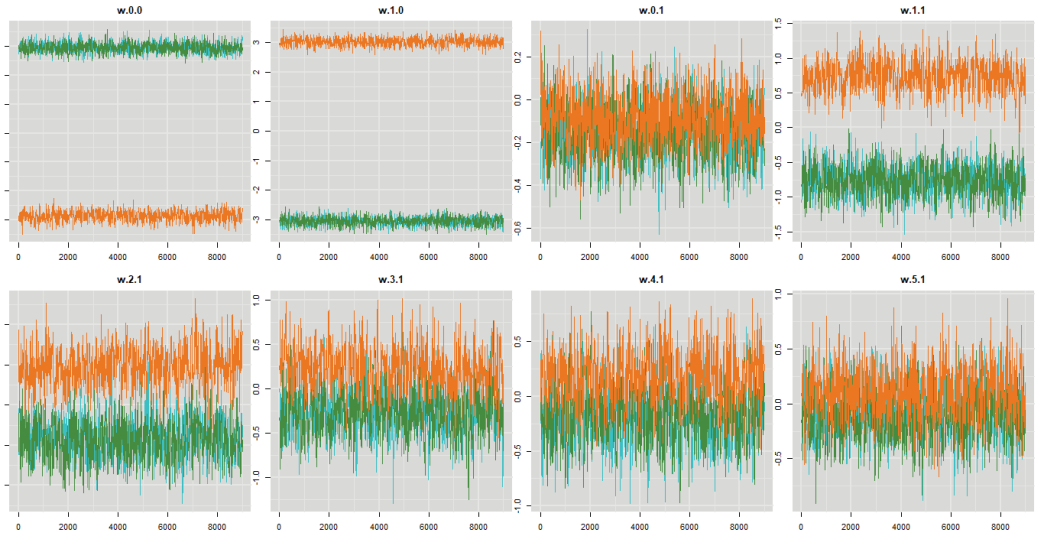
Nas FIGURAS 34 e 35 tem-se respectivamente a distribuição *a posteriori* e as cadeias de Markov para 1 neurônio na camada oculta.

FIGURA 34 – DISTRIBUIÇÃO *A POSTERIORI* COM 1 NEURÔNIO



FONTE: A autora (2020).

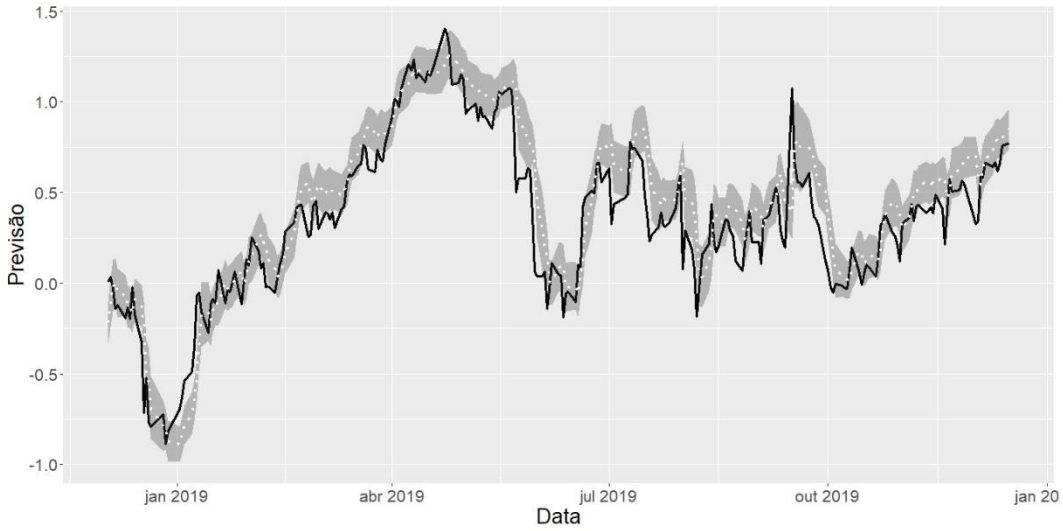
FIGURA 35 - CADEIAS DE MARKOV COM 1 NEURÔNIO



FONTE: A autora (2020).

Na FIGURA 36 tem-se o intervalo com 95% de credibilidade para o período de validação com um neurônio, construído com base na informação de probabilidade *a posteriori*. Apesar de acompanhar o comportamento histórico o intervalo não contempla o verdadeiro valor em um número considerável de pontos. É possível observar também a média do intervalos representada em pontilhado branco.

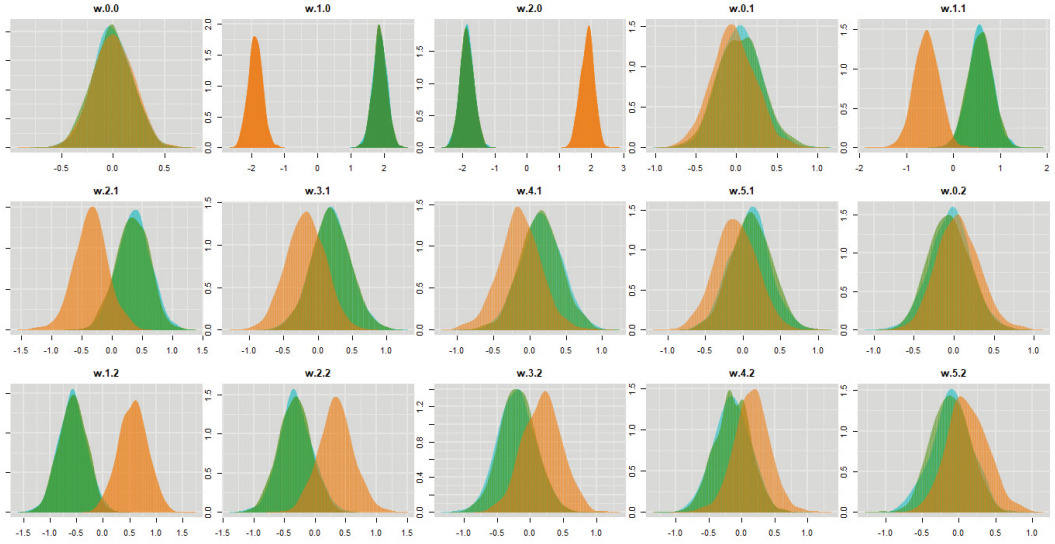
FIGURA 36 - INTERVALO DE CREDIBILIDADE COM 1 NEURÔNIO



FONTE: A autora (2020).

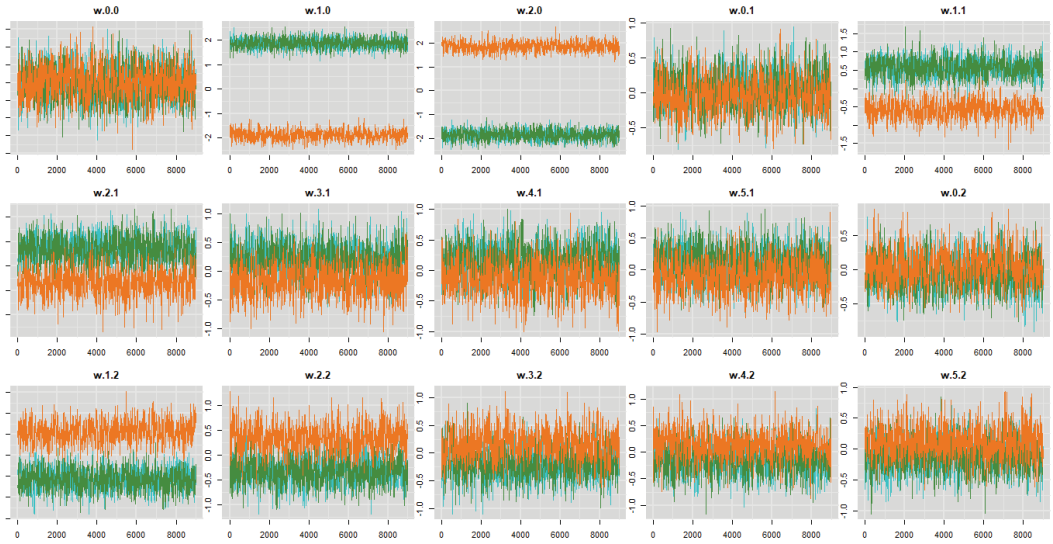
Nas FIGURAS 37 e 38 tem-se a distribuição *a posteriori* e as cadeias de Markov para a configuração de dois neurônios.

FIGURA 37 – DISTRIBUIÇÃO *A POSTERIORI* COM 2 NEURÔNIOS



FONTE: A autora (2020).

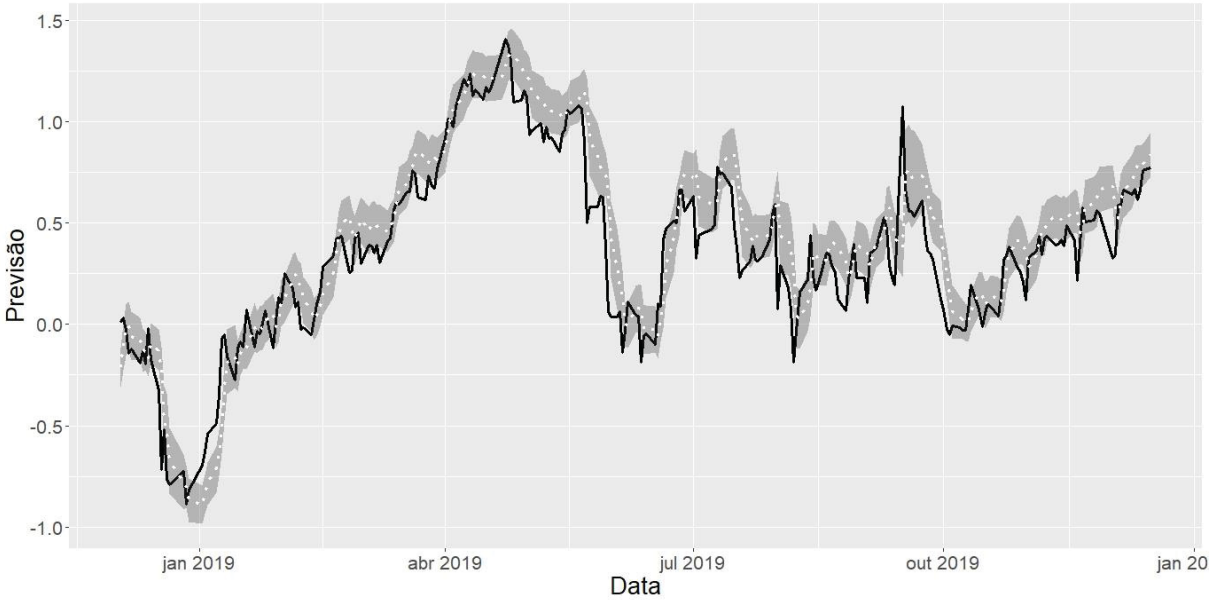
FIGURA 38 - CADEIAS DE MARKOV COM 2 NEURÔNIOS



FONTE: A autora (2020).

Na FIGURA 39 tem-se o intervalo com 95% de credibilidade para dois neurônios na camada oculta da rede. É possível notar que com dois neurônios o resultado é semelhante ao de um neurônio, aparentemente contempla maior número de valores no intervalo, porém, ainda assim contém muitos valores fora deste.

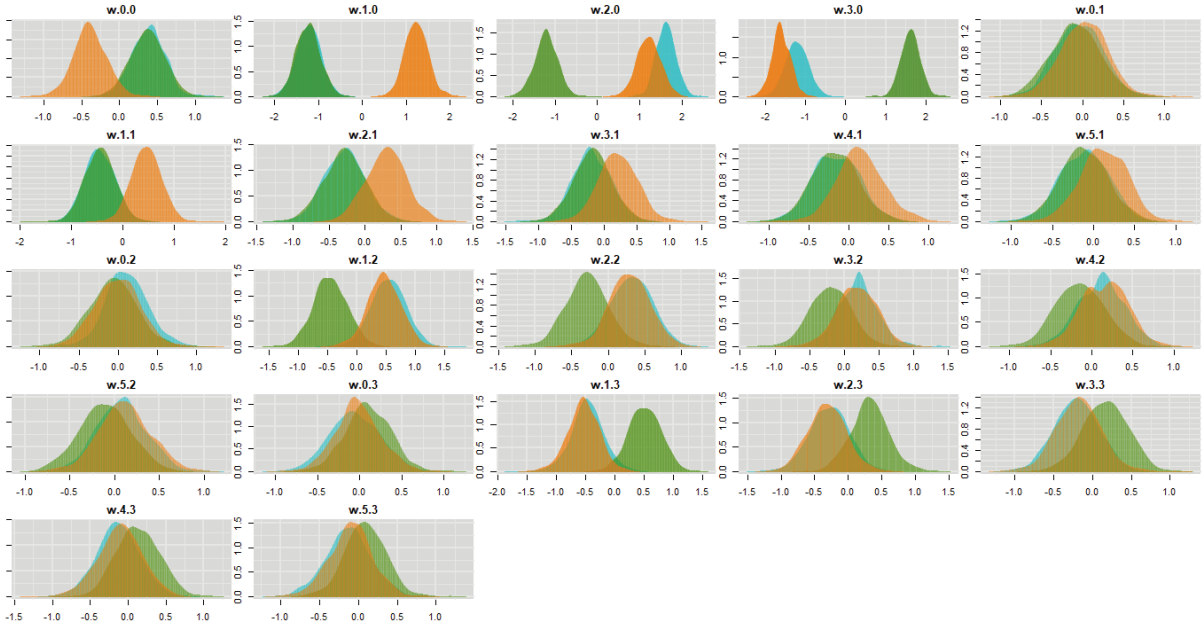
FIGURA 39 - INTERVALO DE CREDIBILIDADE COM 2 NEURÔNIOS



FONTE: A autora (2020).

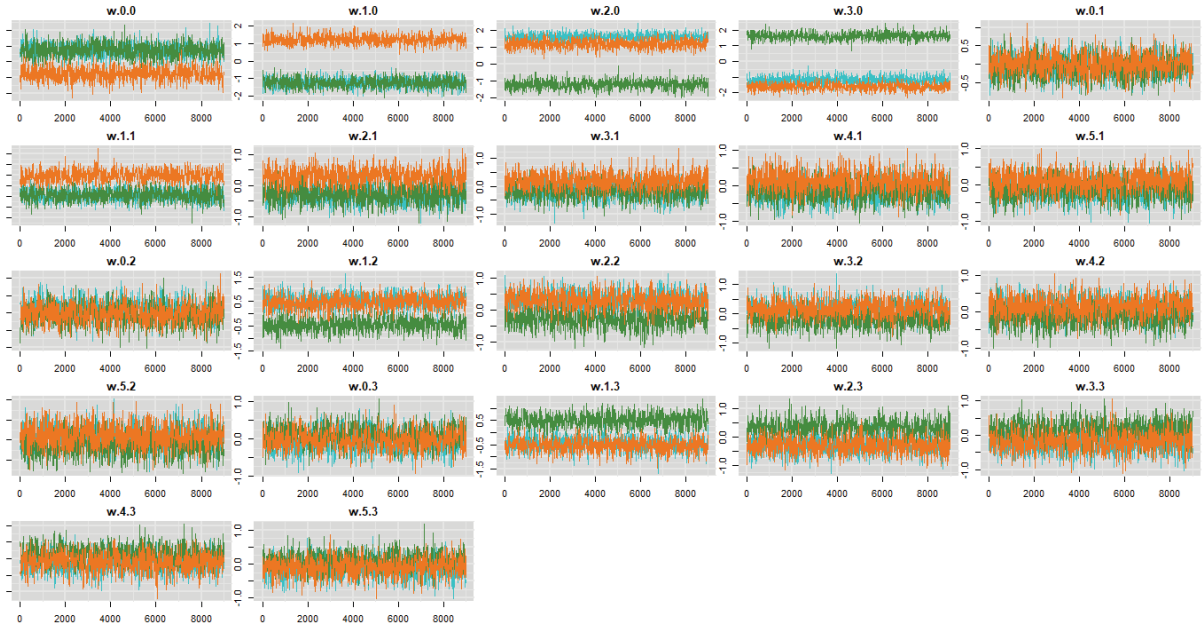
Nas FIGURAS 40 e 41 tem-se a distribuição *a posteriori* e as cadeias de Markov para a configuração de três neurônios.

FIGURA 40 - DISTRIBUIÇÃO *A POSTERIORI* COM 3 NEURÔNIOS



FONTE: A autora (2020).

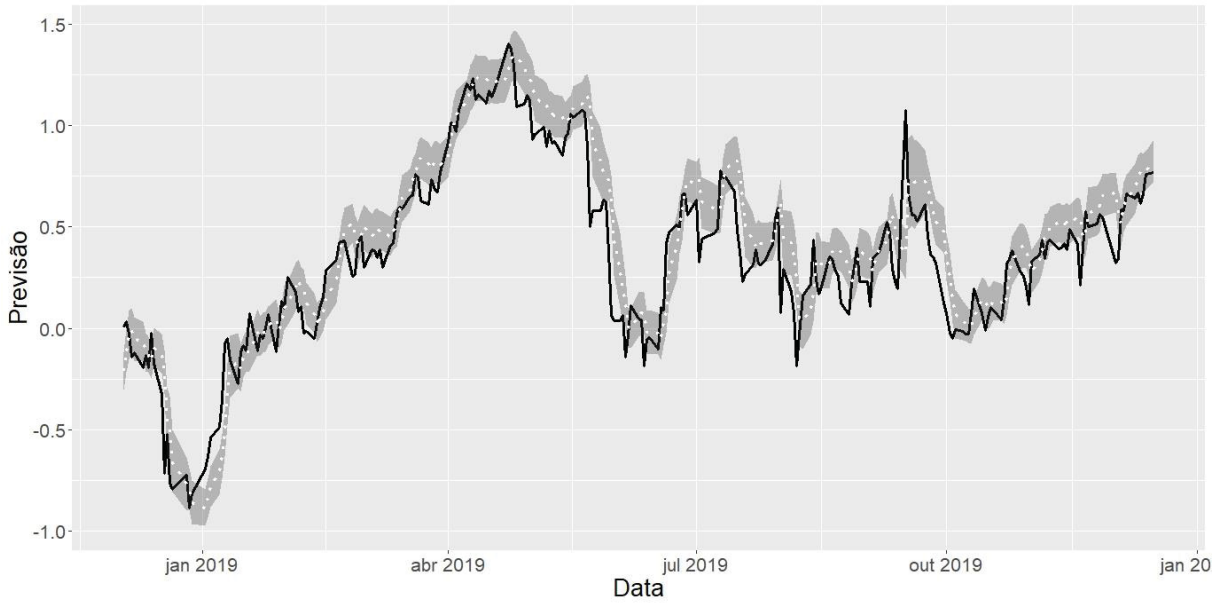
FIGURA 41 - CADEIAS DE MARKOV COM 3 NEURÔNIOS



FONTE: A autora (2020).

Na FIGURA 42 tem-se o intervalo com 95% de credibilidade para três neurônios na camada oculta, que mesmo aparentando estar melhor do que os anteriores ainda sim, possui bastantes pontos fora do intervalo.

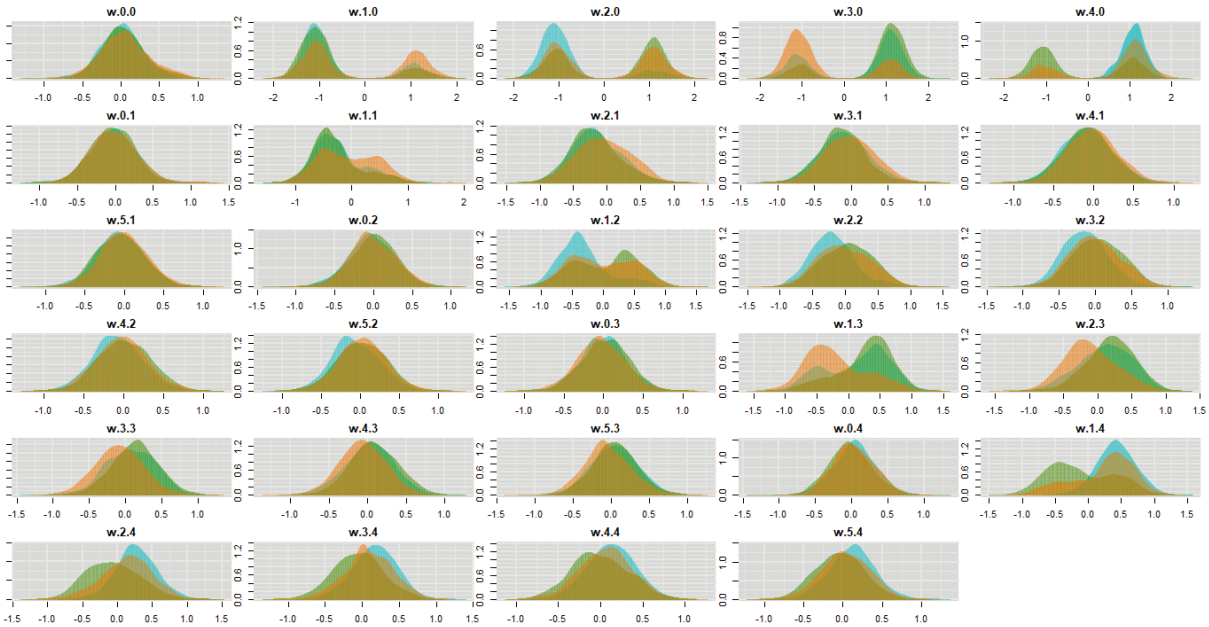
FIGURA 42 - INTERVALO DE CREDIBILIDADE COM 3 NEURÔNIOS



FONTE: A autora (2020).

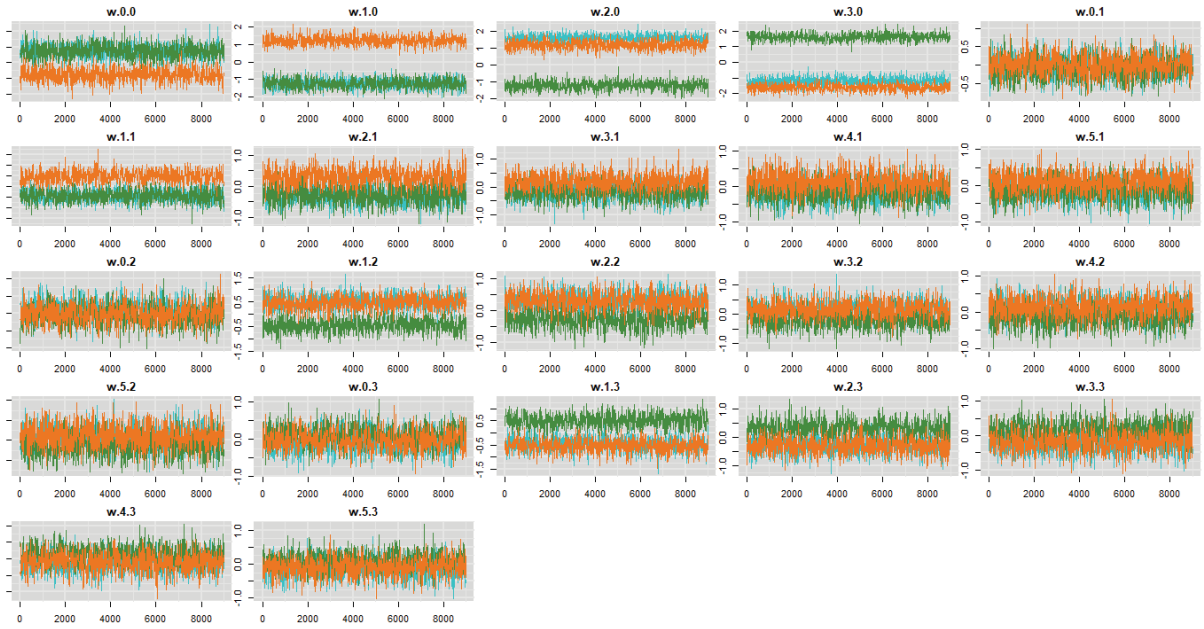
Nas FIGURAS 43 e 44 tem-se a distribuição *a posteriori* e as cadeias de Markov para a configuração de quatro neurônios.

FIGURA 43 - DISTRIBUIÇÃO *A POSTERIORI* COM 4 NEURÔNIOS



FONTE: A autora (2020).

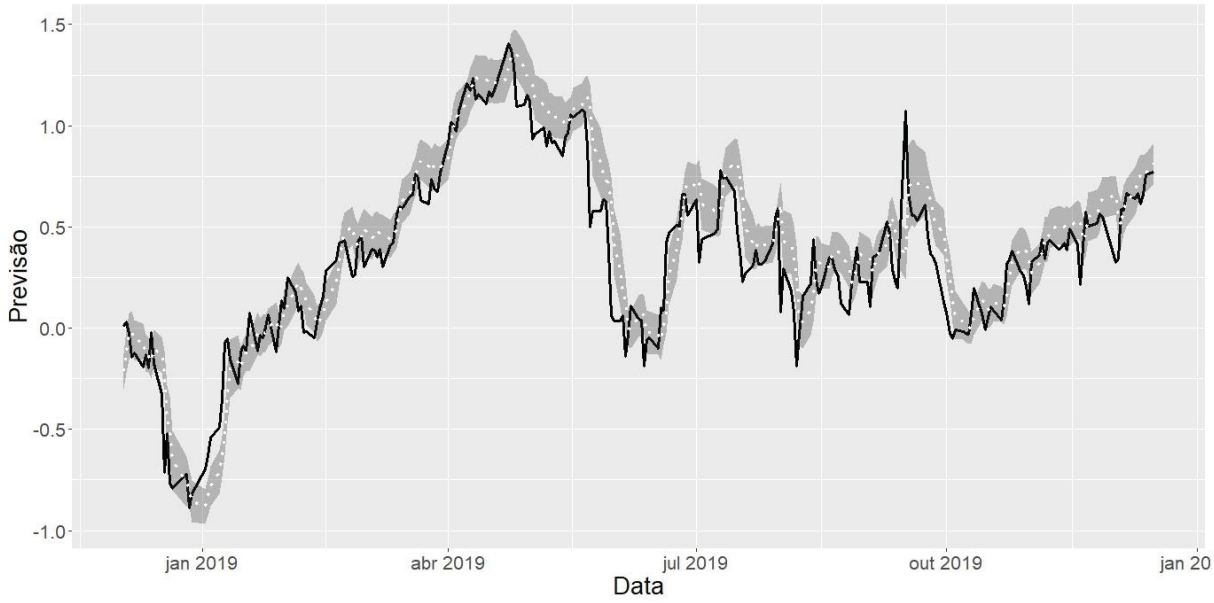
FIGURA 44 - CADEIAS DE MARKOV COM 4 NEURÔNIOS



FONTE: A autora (2020).

Na FIGURA 45 tem-se o intervalo com 95% de credibilidade para quatro neurônios, o qual também não apresenta um bom resultado.

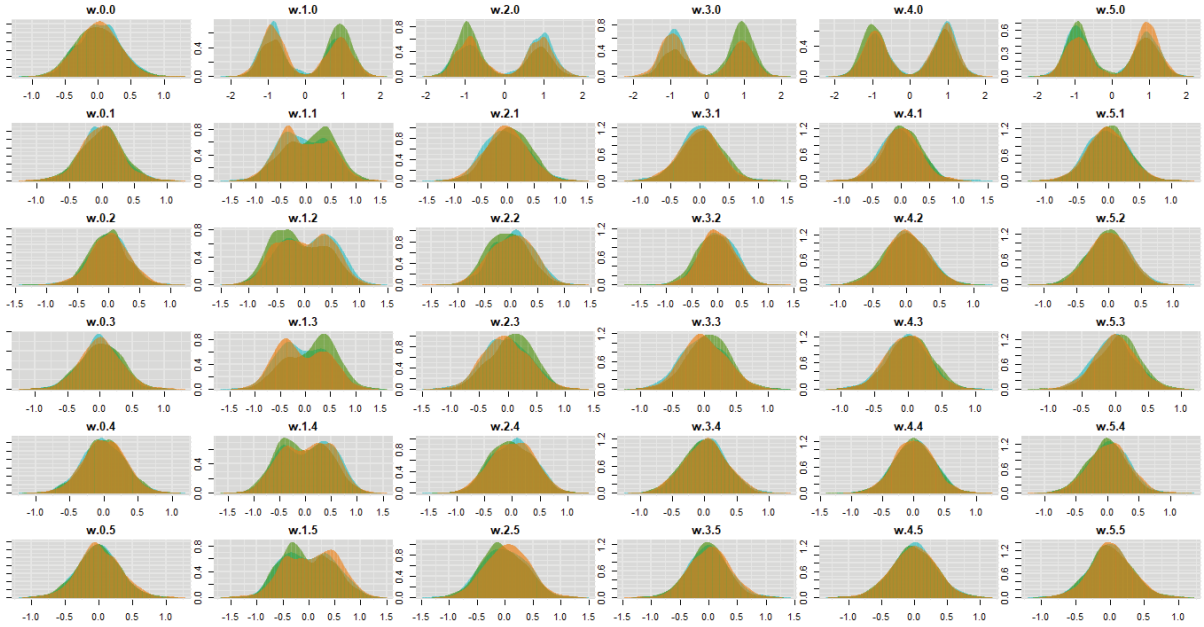
FIGURA 45 - INTERVALO DE CREDIBILIDADE COM 4 NEURÔNIOS



FONTE: A autora (2020).

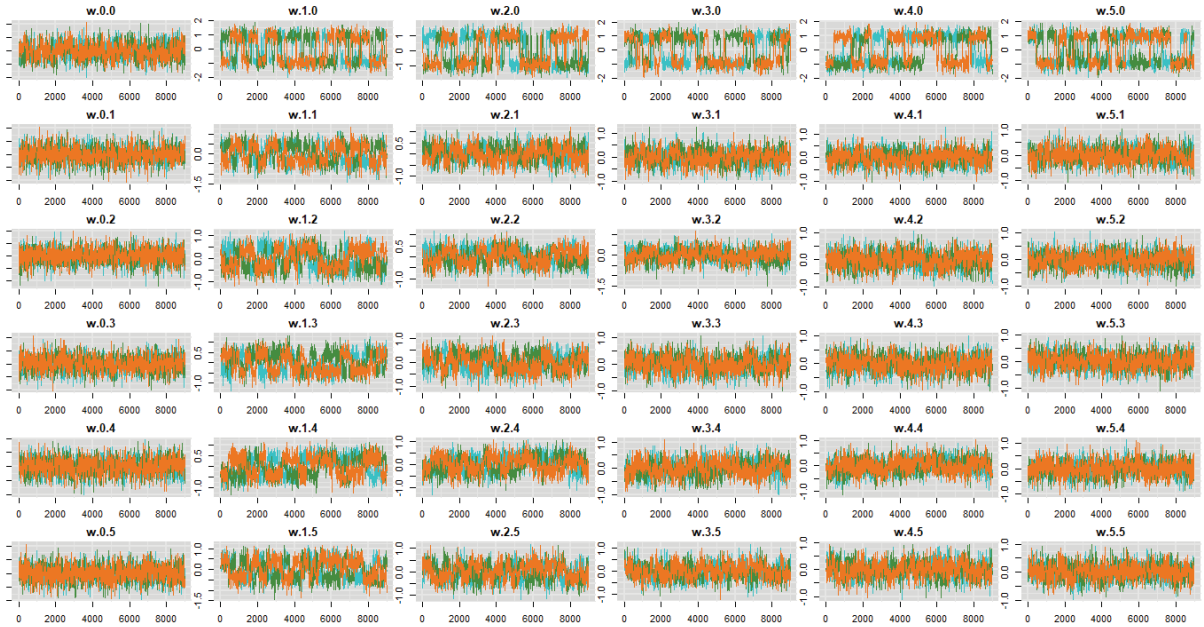
Nas FIGURAS 46 e 47 tem-se a distribuição *a posteriori* e as cadeias de Markov para 5 neurônios.

FIGURA 46 - DISTRIBUIÇÃO *A POSTERIORI* COM 5 NEURÔNIOS



FONTE: A autora (2020).

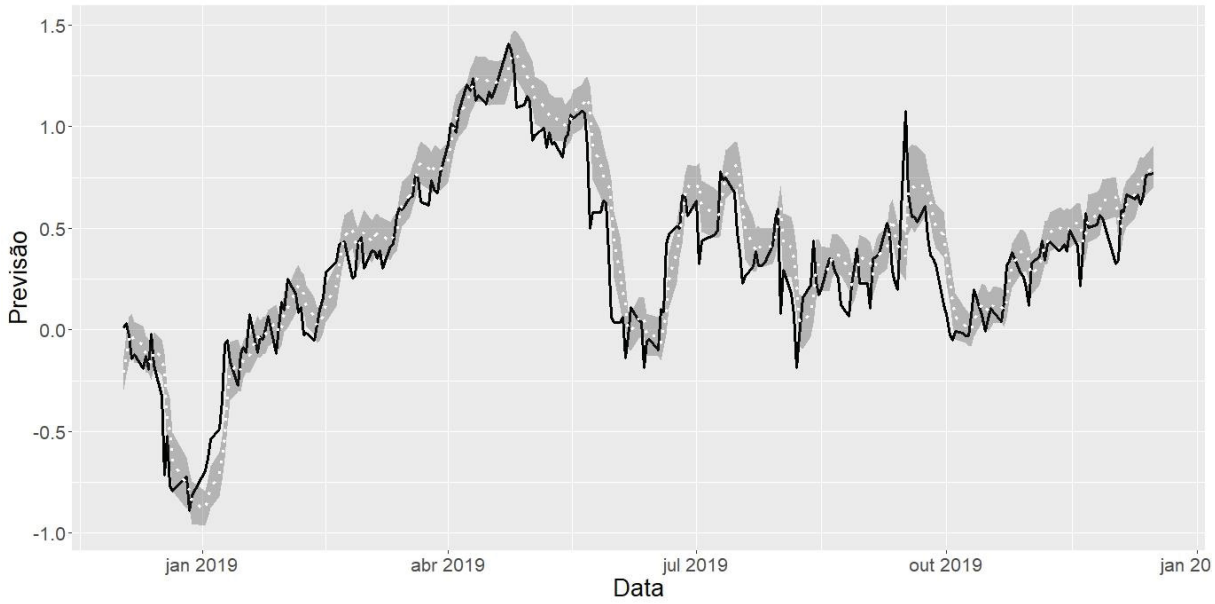
FIGURA 47 - CADEIAS DE MARKOV COM 5 NEURÔNIOS



FONTE: A autora (2020).

Na FIGURA 48 tem-se o intervalo com 95% de credibilidade para cinco neurônios, que apesar de contemplar mais pontos no intervalo ainda possui grande parte fora.

FIGURA 48 - INTERVALO DE CREDIBILIDADE COM 5 NEURÔNIOS



FONTE: A autora (2020).

Para aplicar o modelo baseado no SIC, incialmente foi observada a correlação das variáveis, conforme a TABELA 6, para assim selecionar aquela que faz parte da configuração inicial.

TABELA 6– CORRELAÇÃO COM A VARIÁVEL RESPOSTA

x_1	x_2	x_3	x_4	x_5
0.993	0.988	0.981	0.977	0.971

FONTE: A autora (2020).

A covariável x_1 possui maior correlação com a resposta e foi então selecionada para compor o modelo inicial. Os parâmetros da covariável e do intercepto foram então estimadas pelo algoritmo SA e os parâmetros que ligam o neurônio com a camada de saída e o intercepto desta foram estimados pela regressão linear. O modelo inicial então está na equação (45).

$$\mu_i = \quad 11.76 - 23.42\frac{1}{(1+e^{-(0.01-0.17x_{i1})})} \tag{45}$$

A partir disto, fez-se então o *ranking* do SIC para avaliar se alguma variável entraria no modelo inicial. Conforme os resultados, dados na TABELA 7, no primeiro SIC do primeiro neurônio, x_4 passa a fazer parte do modelo.

TABELA 7– *RANKING* SIC POR NEURÔNIO

	1º Neurônio		2º Neurônio	3º Neurônio
	$1^{o}SIC$	$2^{o}SIC$	$1^{o}SIC$	$1^{o}SIC$
x_2	-4.97	6.55	11.96	14.15
x_3	2.79	7.89	11.88	15.95
x_4	**-9.76**	-	12.00	14.89
x_5	5.39	2.17	11.95	15.87

FONTE: A autora (2020).

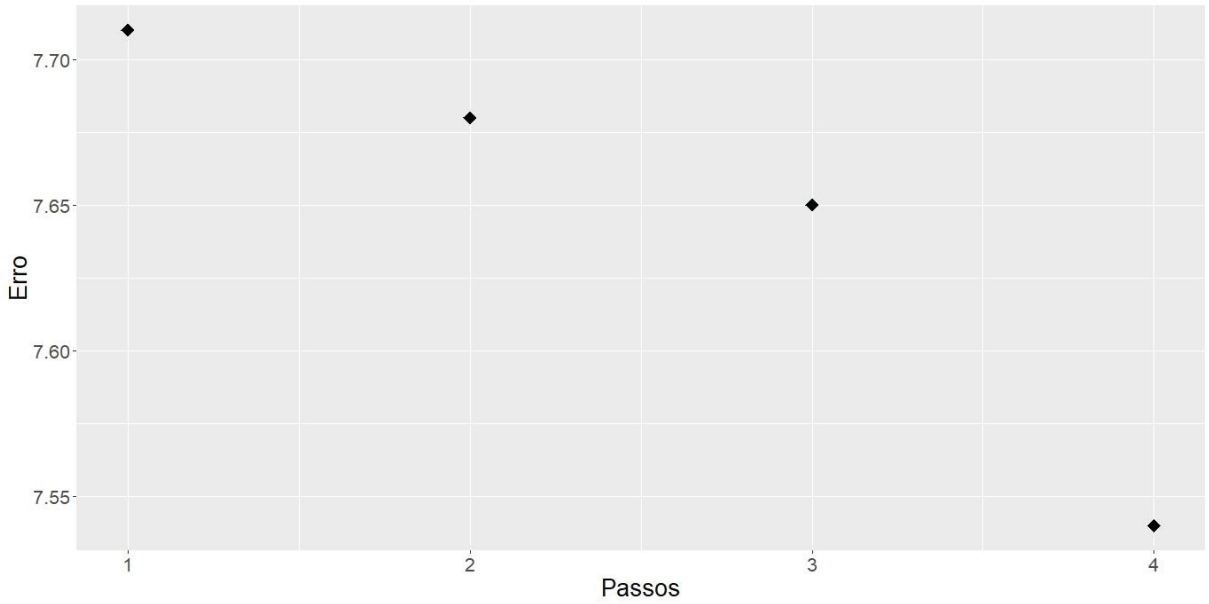
Então novamente são estimados os parâmetros assim como feitos anteriormente e tem-se novo modelo, com o segundo *ranking* SIC tem-se que nenhuma outra variável entra no modelo.

Então, inicia-se o segundo neurônio, com a mesma covariável x_1 e o intercepto, além da estrutura do neurônio um. Nenhuma outra covariável entrou no segundo neurônio, porém, como este possui erro inferior ao modelo anterior então permanece a configuração iniciada. O mesmo ocorreu com o neurônio 3. No neurônio 4 o erro foi maior que o neurônio 3, então este não entrou no modelo. Na equação (46) está o modelo final.

$$\mu_i = 3.20 - 0.19\frac{1}{(1+e^{-(-0.06-2.85x_{i1}-1.725x_{i4})})} - 3.43\frac{1}{(1+e^{-(1.62-1.12x_{i1})})} - 2.89\frac{1}{(1+e^{-(-2.31-1.31x_{i1})})} \quad (46)$$

O erro do modelo foi de 7,54, sendo assim, foi inferior a todos os 150 modelos estimados considerando os algoritmos *slr* , *rprop*+ e *rprop*- com um neurônio na camada oculta. Também obteve melhor resultado do que 48 estimações das 50 do *rprop*+ com 2 neurônios e 49 das 50 do rprop- de 2 neurônios. Com 3 neurônios o *rprop*+ e o *rprop*- foram superiores em 44 estimações cada. Já com 4 e 5 neurônios as 50 estimativas dos algoritmos *rprop*+ e *rprop*- foram melhores. Na FIGURA 49 é possível verificar as melhorias do modelo a cada passo do algoritmo.

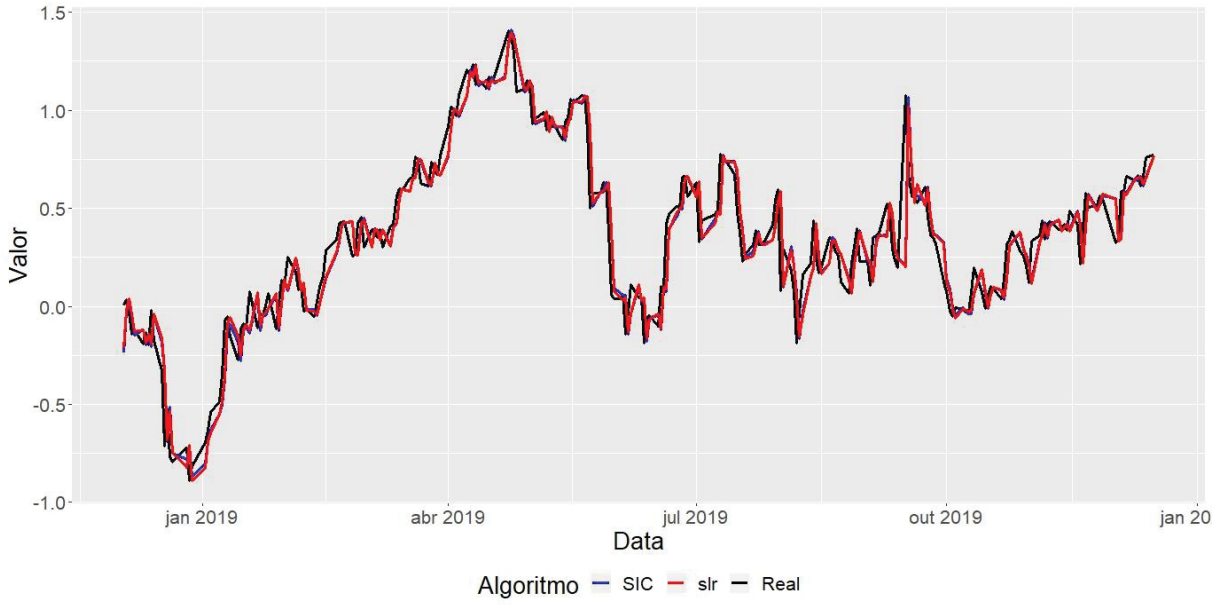
FIGURA 49 - ERROS DO MODELO



FONTE: A autora (2020).

Na FIGURA 50 está a previsão no período de validação do modelo proposto juntamente com o *slr*, que foi o algoritmo que melhor conseguiu prever considerando a validação. Pode-se perceber que os resultados são semelhantes, o erro de previsão do algoritmo proposto foi de 2,30; o mesmo obtido pelo *slr* na validação.

FIGURA 50 - PREVISÃO COM SIC



FONTE: A autora (2020).

É importante ressaltar que o algoritmo *slr* obteve dificuldade em convergência para outras configurações de rede além da de um neurônio, sendo assim, fica difícil comparar o modelo proposto com este que não convergiu em todas as configurações de rede.

São também avaliados os melhores modelos obtidos nos dois outros algoritmos *rprop+* e *rprop-*, cada qual com 36 parâmetros, os quais convergiram para todas as configurações de rede. Com o modelo proposto tem-se uma estrutura de 10 parâmetros, sendo foi também avaliadas todas as estruturas de neurônios, sem problemas de convergência.

4.2 ESTUDO DE CASO 2: EXPORTAÇÃO NO SETOR ALIMENTÍCIO

Para o estudo de caso de exportação no setor alimentício foram consideradas 5 covariáveis, sendo elas IPCA de alimentos e bebidas (x_1), taxa de câmbio R$/US$ (x_2), câmbio contratado em exportações (x_3), imposto sobre a importação (x_4) e exportações (x_5) e tem-se 220 informações de cada variável para o treinamento. Assim como no estudo de caso 1, foram avaliadas as arquiteturas de rede com 1 a 5 neurônios na camada oculta de uma rede MLP de função de ativação sigmóide logística.

Os dados foram treinados 50 vezes por cada algoritmo. Na TABELA 8 tem-se os menores erros obtidos nas previsões geradas. Para o algoritmo *rprop+* o menor erro foi de 2,04, obtido com 7.686 iterações. No *rprop-*, com 9.887 iterações, o menor erro foi de 2,11. O *slr* com 46.625 iterações o menor erro foi de 1,99.

TABELA 8 – MENORES ERROS DOS ALGORITMOS NO ESTUDO DE CASO 2

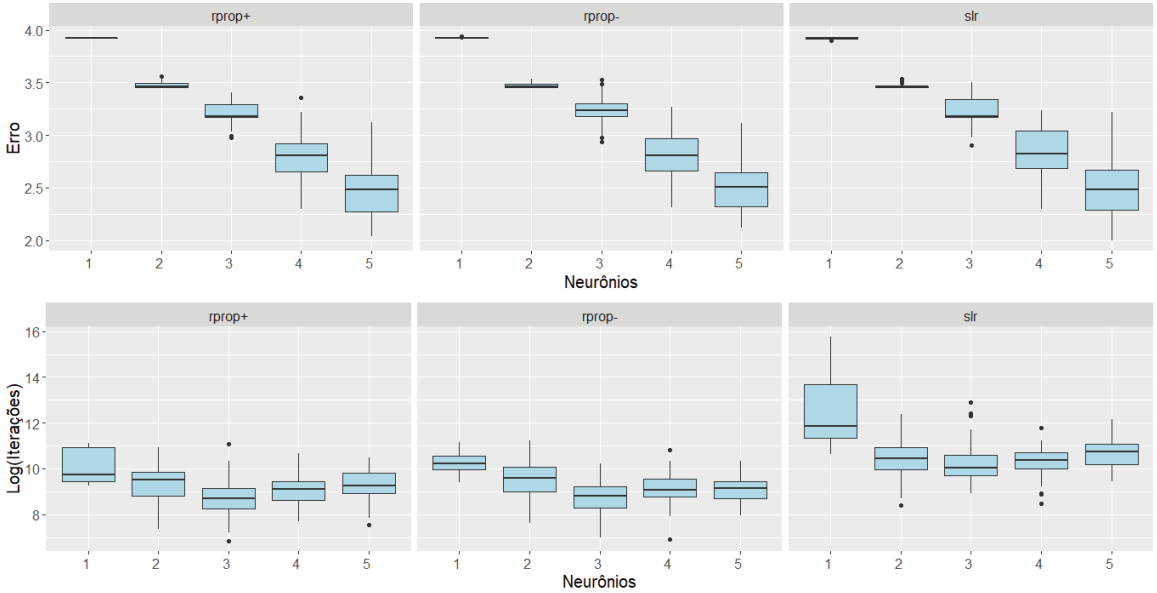
	rprop+	***rprop-***	***slr***
1 neurônio	3.92	3.92	3.90
2 neurônios	3.45	3.45	3.45
3 neurônios	2.97	2.93	2.90
4 neurônios	2.30	2.31	2.30
5 neurônios	**2.04**	**2.11**	**1.99**

FONTE: A autora (2020).

Na FIGURA 51 é possível verificar que quanto maior o número de neurônios menor erro obteve o algoritmo. Também que o algoritmo *slr* precisou de maior volume

de iterações que o *rprop*+ e o *rprop*-, os quais com menos neurônios precisaram de mais iterações.

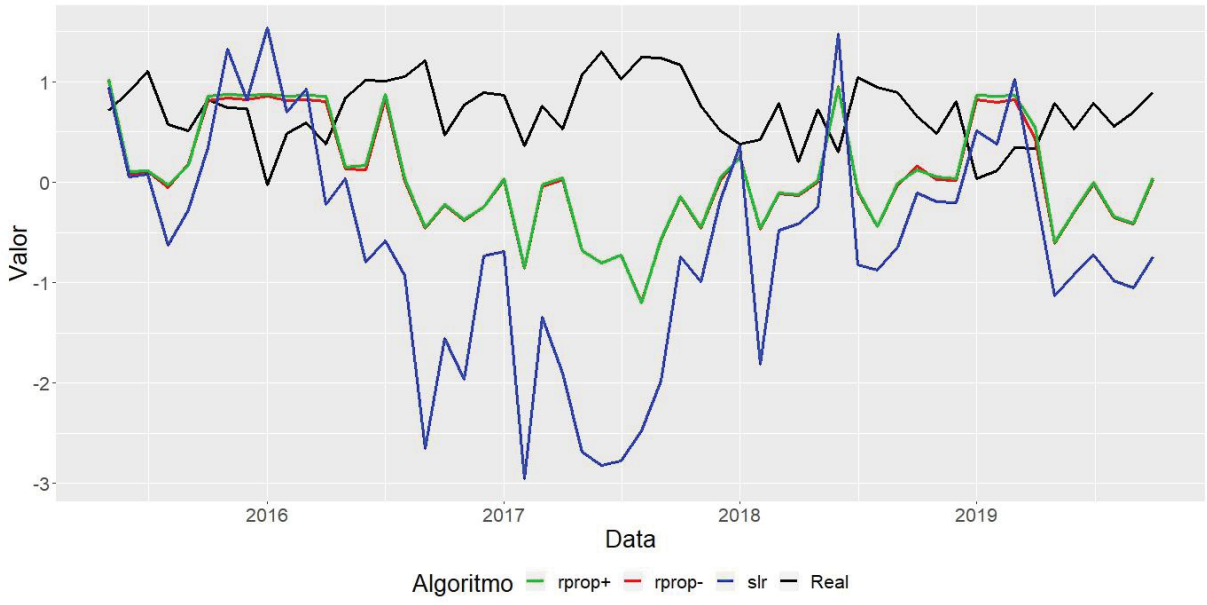
FIGURA 51 – ERROS E ITERAÇÕES DOS ALGORITMOS TRADICIONAIS



FONTE: A autora (2020).

O erro obtido no treinamento com os algoritmos foi de 3,02 com o *rprop*+, 2,75 com o *rprop*- e 10,03 com o *slr* . Ainda assim, é perceptível, pela FIGURA 52, que nenhum algoritmo obteve bom desempenho.

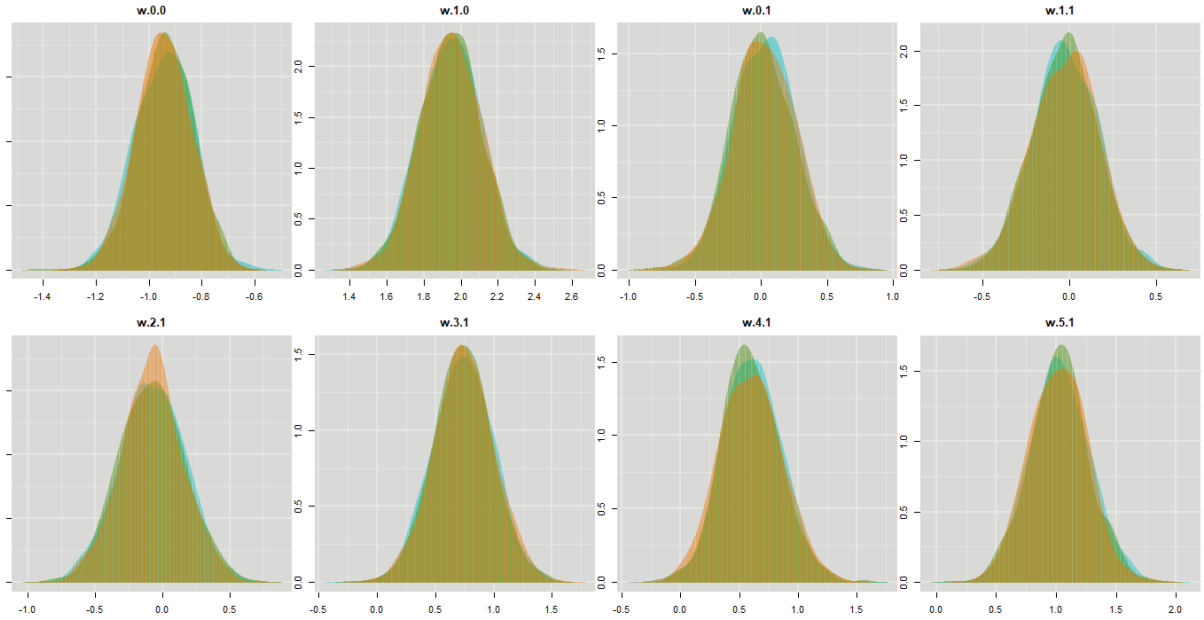
FIGURA 52 - PREVISÃO DOS ALGORITMOS TRADICIONAIS



FONTE: A autora (2020).

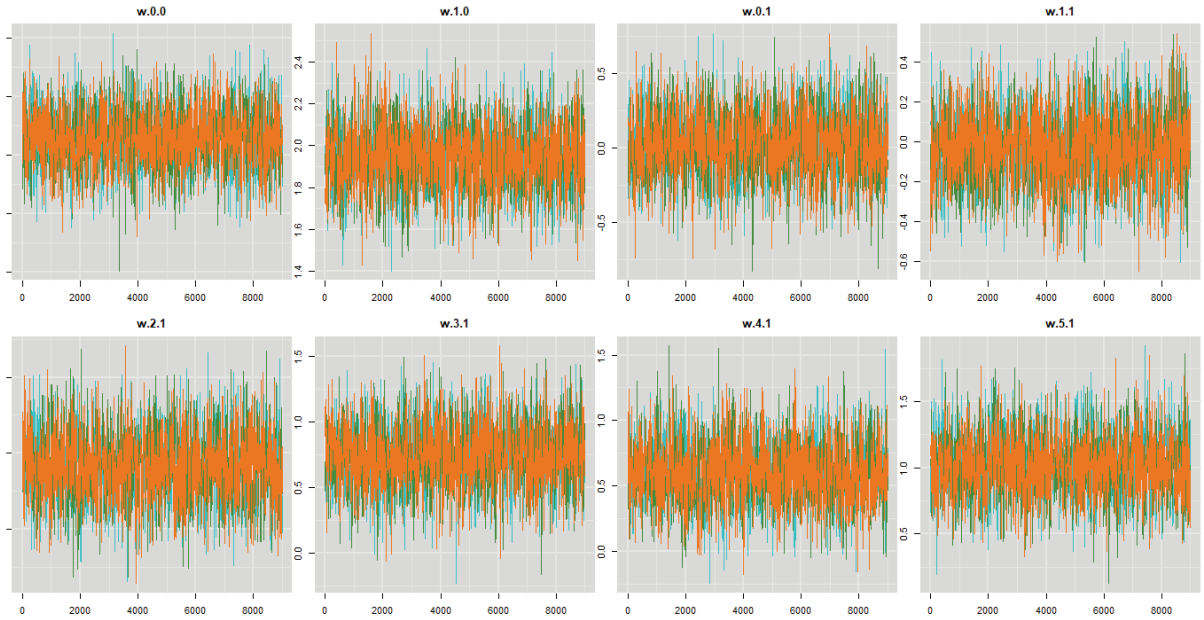
Nas FIGURAS 53 e 54 tem-se respectivamente a distribuição *a posteriori* e as cadeias de Markov para 1 neurônio na camada oculta.

FIGURA 53 - DISTRIBUIÇÃO *A POSTERIORI* COM 1 NEURÔNIO



FONTE: A autora (2020).

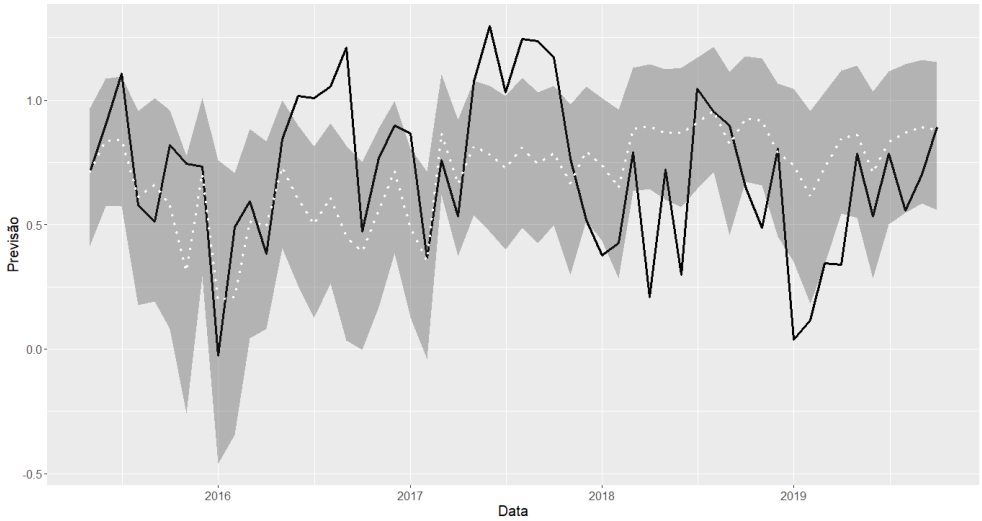
FIGURA 54 - CADEIAS DE MARKOV COM 1 NEURÔNIO



FONTE: A autora (2020).

Para o caso do treinamento com 1 neurônio o intervalo de credibilidade está na FIGURA 55. É perceptivel que para este estudo de caso o resultado da inferência Bayesiana não foi satisfatório.

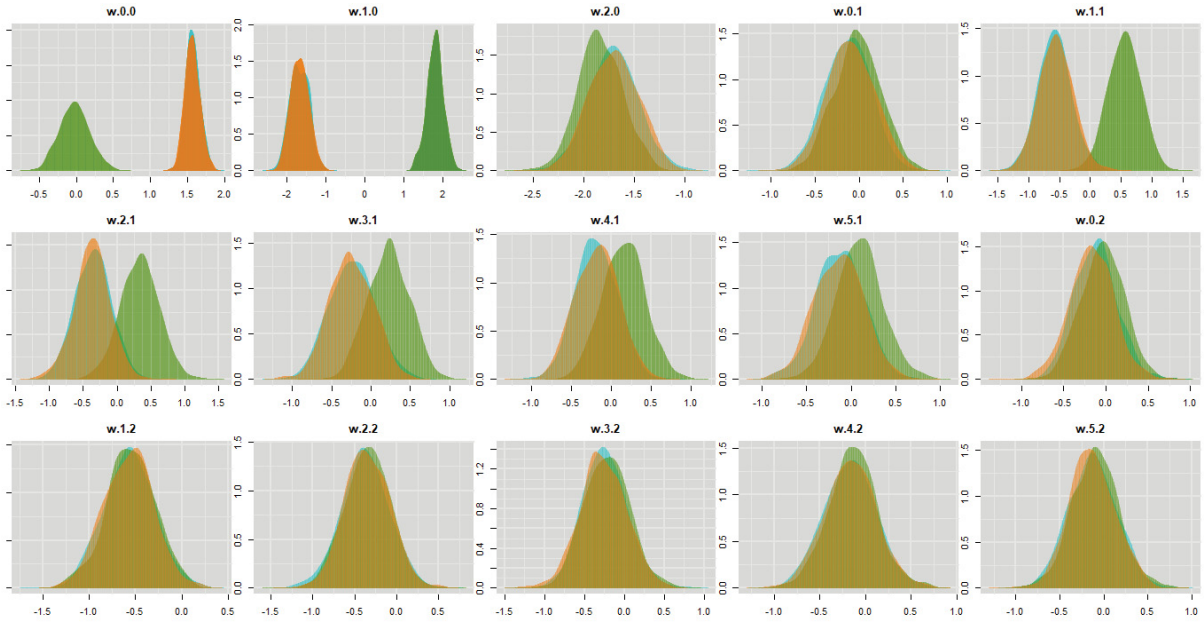
FIGURA 55 - INTERVALO DE CREDIBILIDADE COM 1 NEURÔNIO



FONTE: A autora (2020).

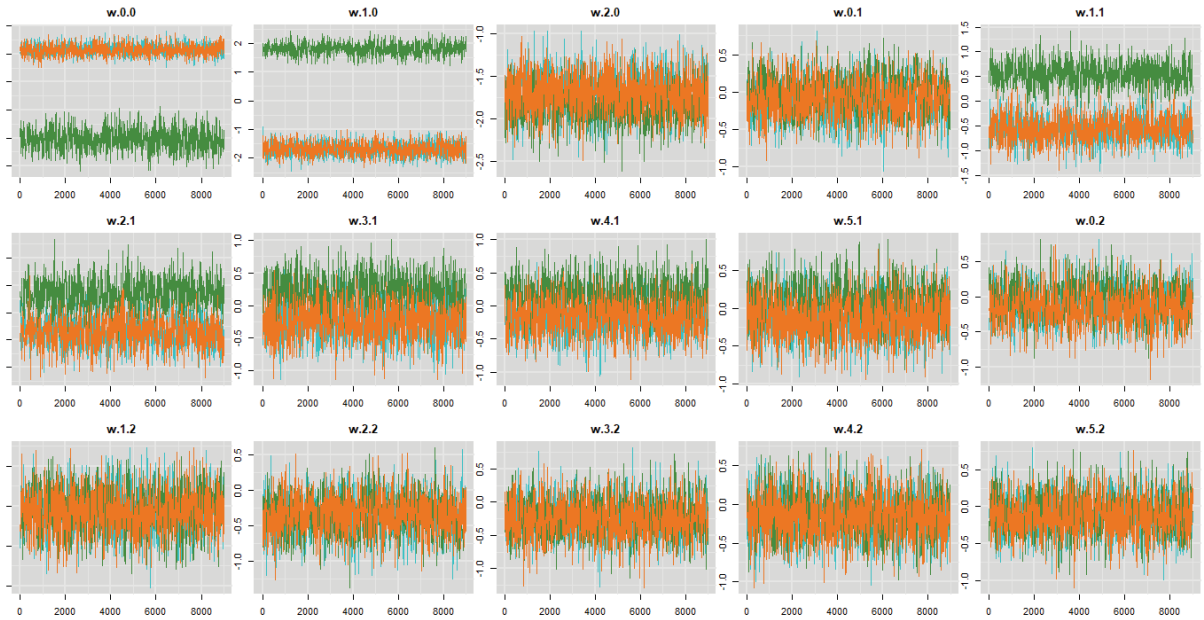
Para o caso do treinamento com 2 neurônios nas FIGURAS 56 e 57 tem-se a distribuição *a posteriori* e as cadeias de Markov.

FIGURA 56 - DISTRIBUIÇÃO *A POSTERIORI* COM 2 NEURÔNIOS



FONTE: A autora (2020).

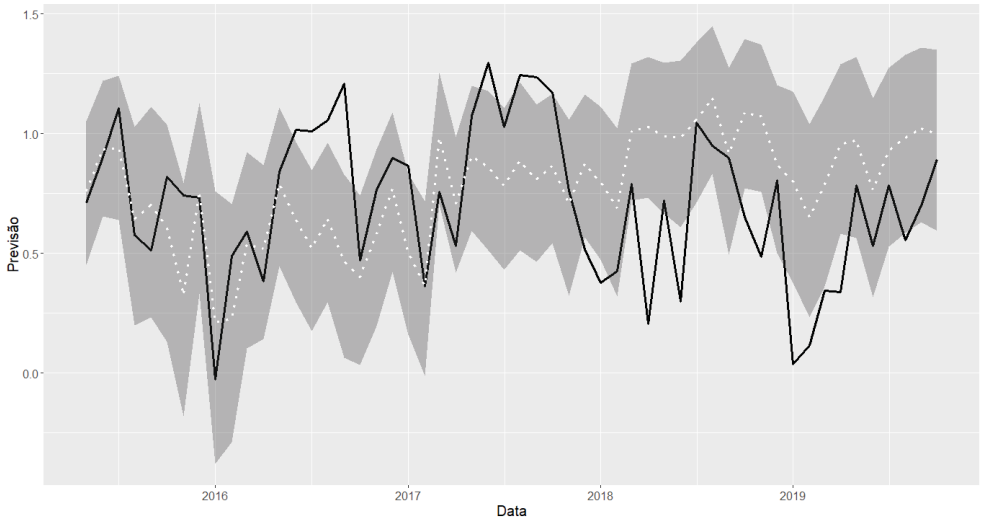
FIGURA 57 - CADEIAS DE MARKOV COM 2 NEURÔNIOS



FONTE: A autora (2020).

Para o caso do treinamento com 2 neurônio o intervalo de credibilidade está na FIGURA 58.

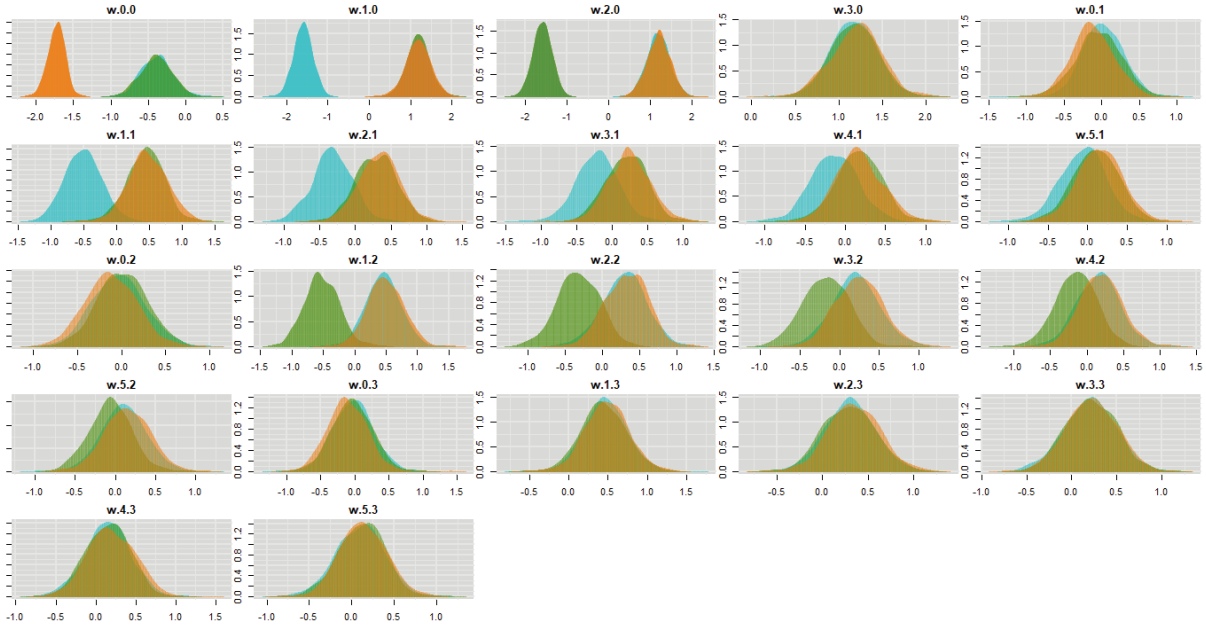
FIGURA 58 - INTERVALO DE CREDIBILIDADE COM 2 NEURÔNIOS



FONTE: A autora (2020).

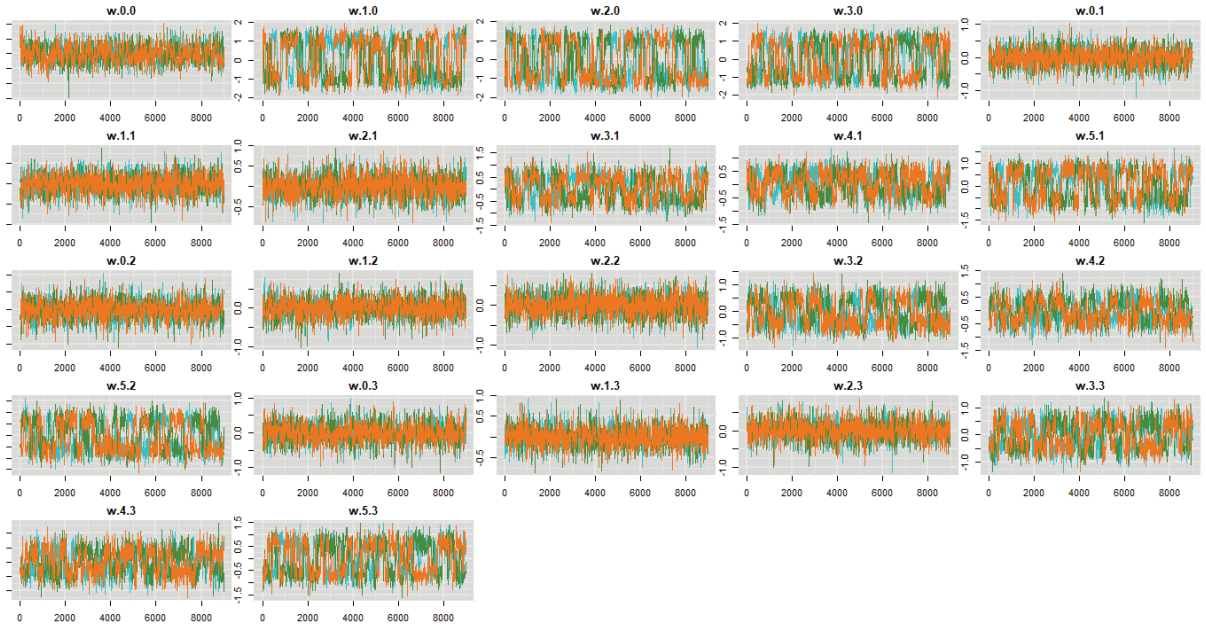
No treinamento com 3 neurônios a distribuição *a posteriori* e as cadeias de Markov estão nas FIGURAS 59 e 60.

FIGURA 59 - DISTRIBUIÇÃO *A POSTERIORI* COM 3 NEURÔNIOS



FONTE: A autora (2020).

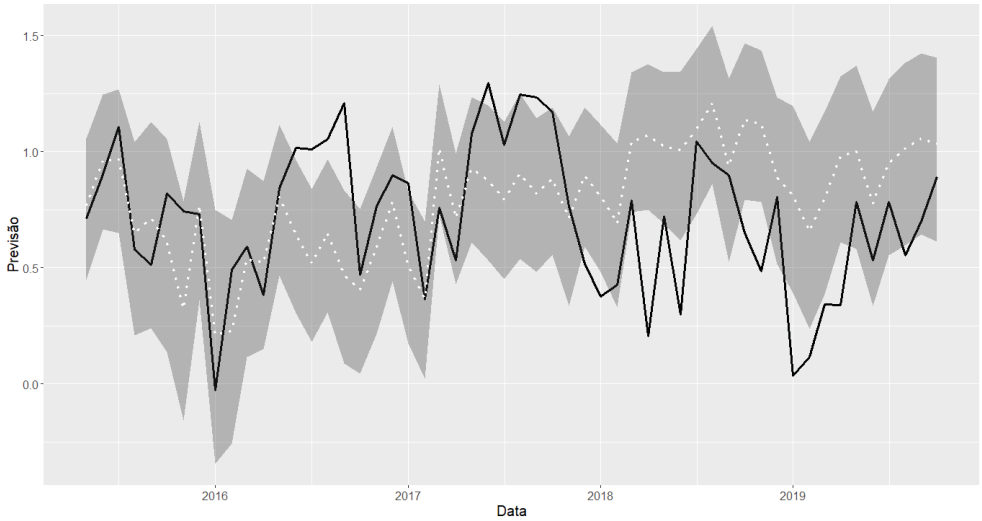
FIGURA 60 - CADEIAS DE MARKOV COM 3 NEURÔNIOS



FONTE: A autora (2020).

Para o caso do treinamento com 3 neurônios o intervalo de credibilidade está na FIGURA 61.

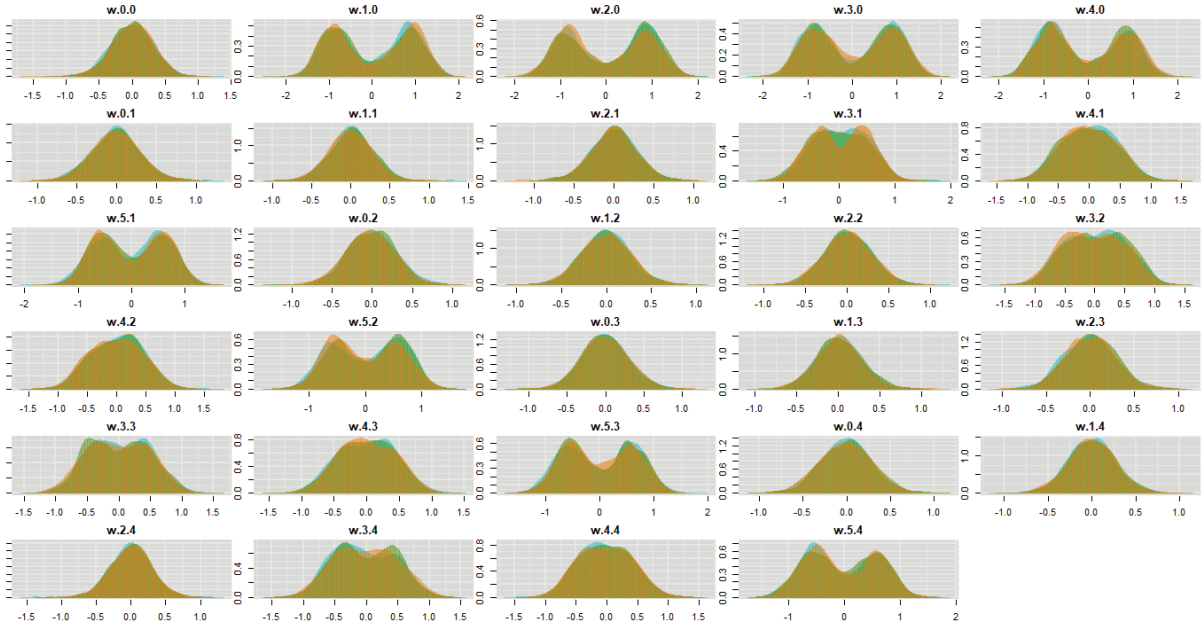
FIGURA 61 - INTERVALO DE CREDIBILIDADE COM 3 NEURÔNIOS



FONTE: A autora (2020).

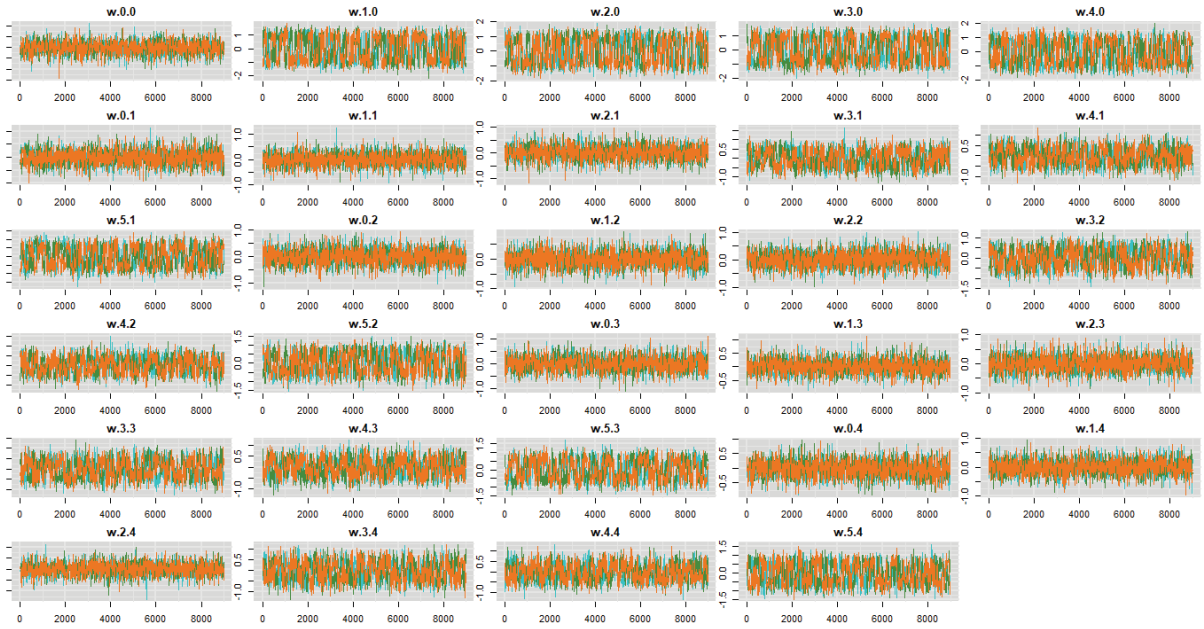
Com 4 neurônios na camada oculta nas FIGURAS 62 e 63 pode-se observar as distribuições *a posteriori* e as cadeias de Markov.

FIGURA 62 - DISTRIBUIÇÃO *A POSTERIORI* COM 4 NEURÔNIOS



FONTE: A autora (2020).

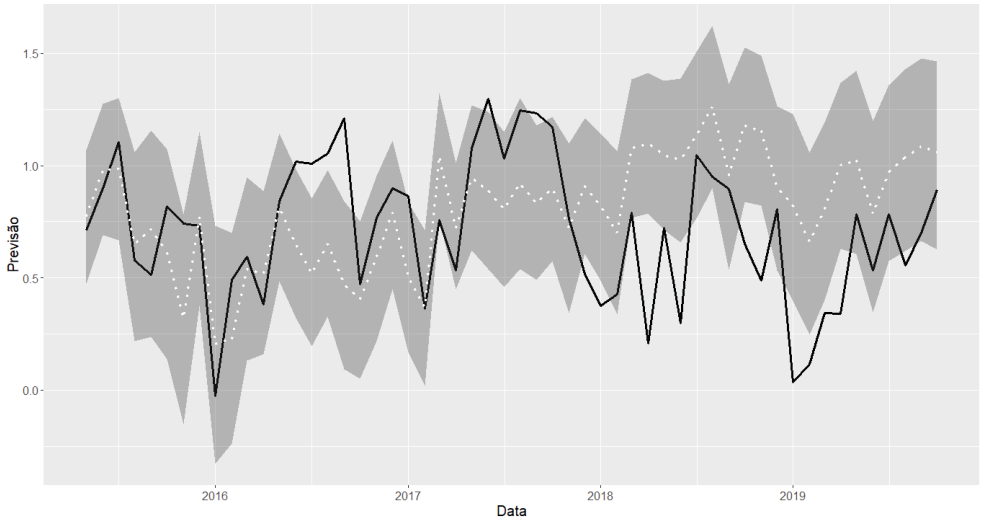
FIGURA 63 - CADEIAS DE MARKOV COM 4 NEURÔNIOS



FONTE: A autora (2020).

Para o caso do treinamento com 4 neurônios o intervalo de credibilidade está na FIGURA 64.

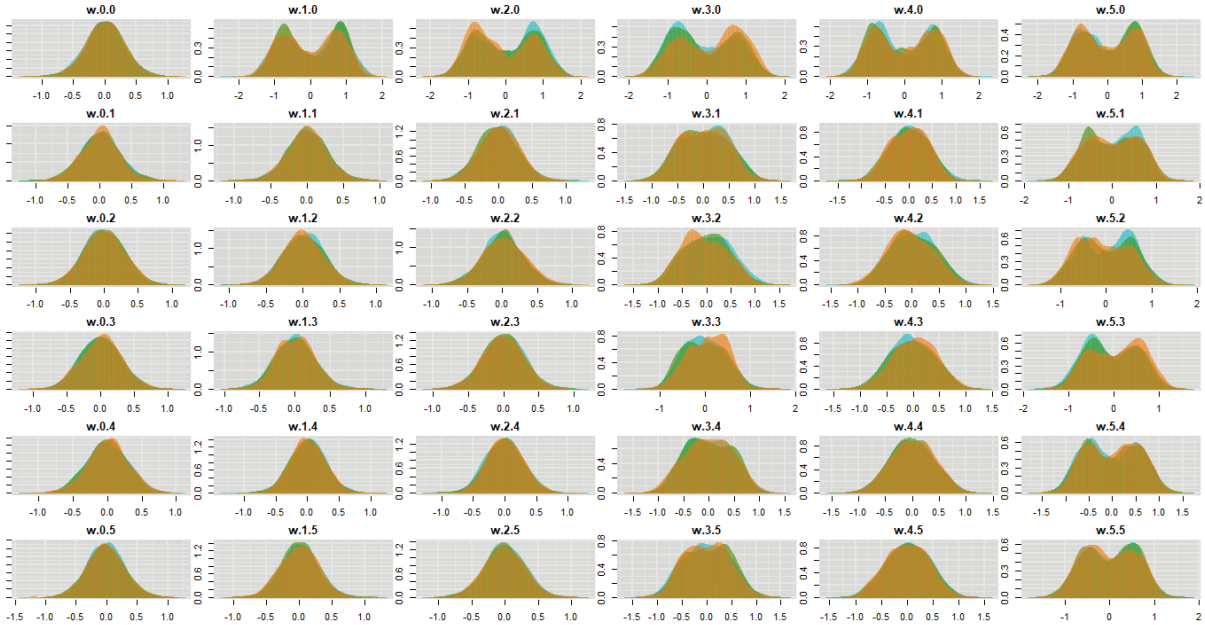
FIGURA 64 - INTERVALO DE CREDIBILIDADE COM 4 NEURÔNIOS



FONTE: A autora (2020).

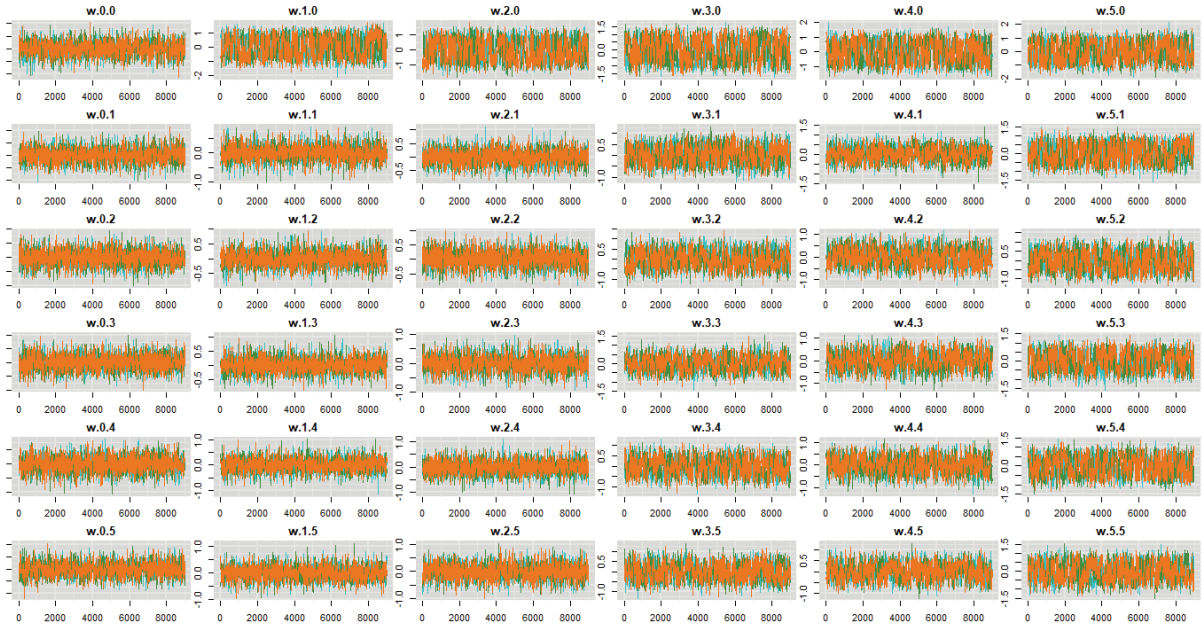
Para o caso do treinamento com 5 neurônios nas FIGURAS 65 e 66 estão a distribuição *a posteriori* e as cadeias de Markov.

FIGURA 65 - DISTRIBUIÇÃO *A POSTERIORI* COM 5 NEURÔNIOS



FONTE: A autora (2020).

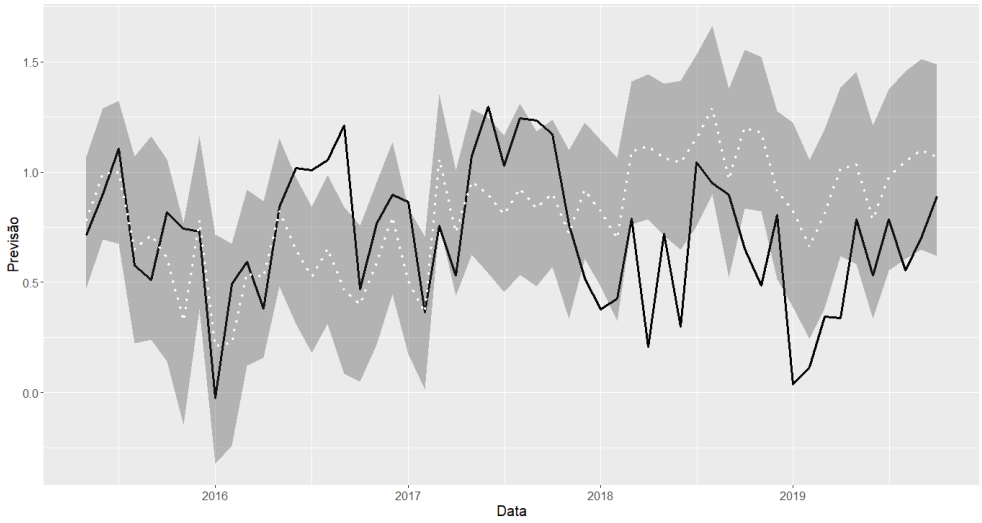
FIGURA 66 - CADEIAS DE MARKOV COM 5 NEURÔNIOS



FONTE: A autora (2020).

O intervalo de credibilidade com o treinamento com 5 neurônios está na FIGURA 67.

FIGURA 67 - INTERVALO DE CREDIBILIDADE COM 5 NEURÔNIOS



FONTE: A autora (2020).

Para aplicar o modelo baseado no SIC, inicialmente foi observada a correlação das variáveis, conforme TABELA 9, para assim selecionar aquela que faz parte da configuração inicial.

TABELA 9– CORRELAÇÃO COM A VARIÁVEL RESPOSTA

x_1	x_2	x_3	x_4	x_5
0.07	-0.04	0.91	0.86	0.98

FONTE: A autora (2020).

A covariável x_5 possui maior correlação com a resposta e foi então selecionada para compor o modelo inicial. Os parâmetros da covariável e do intercepto foram então estimadas pelo algoritmo SA e os parâmetros que ligam o neurônio com a camada de saída e o intercepto desta foram estimados pela regressão linear. O modelo inicial então está na equação (47).

$$\mu_i = \quad 4.28 - 9.24\frac{1}{(1+e^{-(-0.17-0.45x_{i5})})} \tag{47}$$

A partir disto, fez-se então o *ranking* do SIC para avaliar se alguma variável entraria no modelo inicial. Conforme os resultados dados na TABELA 10, pelo primeiro SIC do primeiro neurônio, x_2 passa a fazer parte do modelo.

TABELA 10– *RANKING SIC* PRIMEIRO MODELO

	1º Neurônio		2º Neurônio
	$1^{o}SIC$	$2^{o}SIC$	$1^{o}SIC$
x_1	5.05	7.99	11.20
x_2	**-2.88**	-	10.75
x_3	13.84	26.99	36.04
x_4	13.74	23.92	36.04

FONTE: A autora (2020).

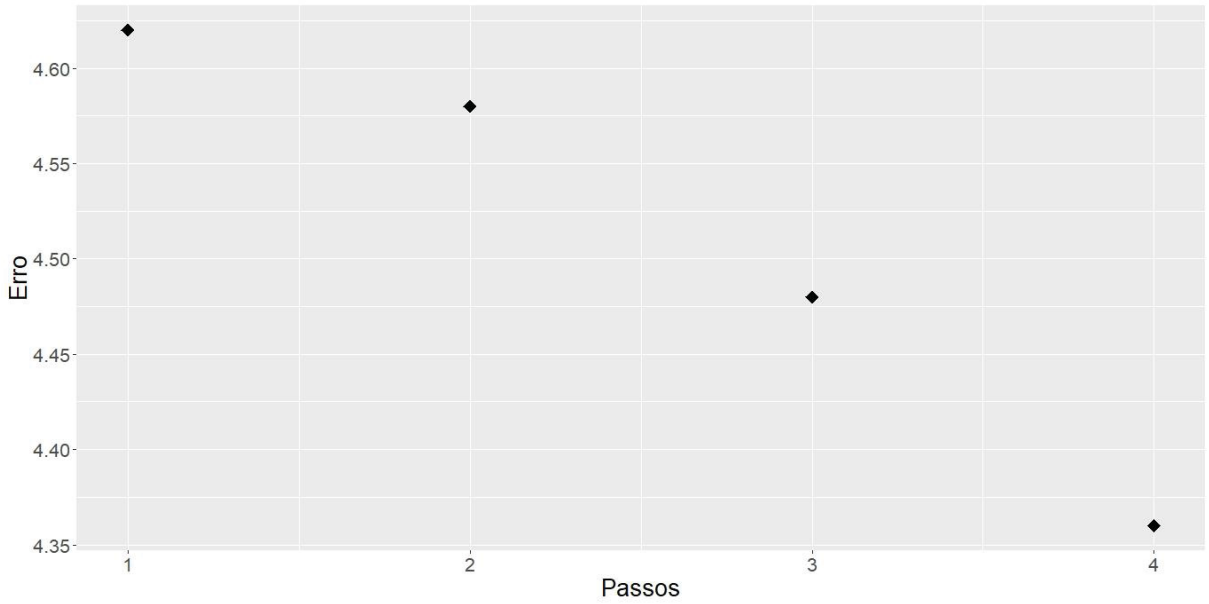
Então novamente são estimados os parâmetros assim como feitos anteriormente, conforme o segundo resultado do *ranking* do SIC não entra mais variável no primeiro neurônio e então é iniciada a avaliação para o segundo neurônio, o qual obteve erro inferior ao modelo anterior. Com a etapa do *ranking SIC*, nenhuma covariável entrou no modelo. Verificou-se então o neurônio três, o qual obteve maior erro e portanto o modelo final está em (48), com erro de 4,48.

$$\mu_i = \quad 3{,}01 + 0{,}12\frac{1}{(1+e^{-(-2{,}59+10{,}10x_{i2}+6{,}89x_{i5})})} - 6.60\frac{1}{(1+e^{-(-0{,}20-0{,}64x_{i5})})} \qquad (48)$$

Se comparado com os erros obtidos nos 50 treinamentos dos algoritmos tradicionais, com um neurônio *slr*, *rprop+* e *rprop-* obtiveram 10 treinamentos com menor erro do que o modelo proposto cada. Com dois neurônios cada um obteve 17 treinamentos com melhores resultados, com 3 foram 24 com 4 foram 31 e com 5 foram 38.

Na FIGURA 68 é possível verificar as melhorias do modelo a cada passo do algoritmo.

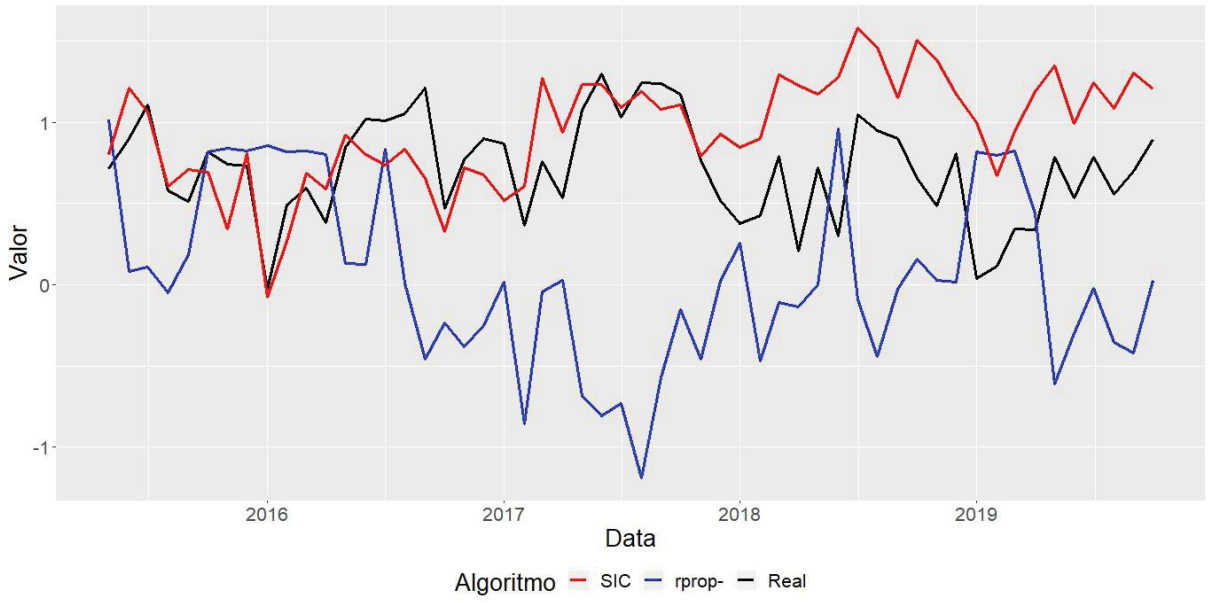
FIGURA 68 - ERROS DO MODELO PROPOSTO



FONTE: A autora (2020).

A previsão com o modelo proposto está na FIGURA 69 assim como a previsão do *rprop-*, que obteve menor erro na validação.

FIGURA 69 - PREVISÃO COM O SIC



FONTE: A autora (2020).

O erro para o modelo SIC no período de validação foi de 5,39; superior ao do rprop- de 2,75. Ainda assim, é possível notar que ambos obtiveram dificuldade em prever o comportamento da série.

5 CONSIDERAÇÕES FINAIS

O objetivo da presente pesquisa foi estudar a estrututa da rede neural *Multilayer Perceptron* a fim de melhor interpretá-la, além de aplicar em dados simulados e posteriormente em reais com três abordagens de aprendizado: métodos tradicionais, que são contemplados em pacotes do *software* R, método da Inferência Bayesiana e por fim uma abordagem proposta de modelo utilizando o *Score Information Criteira*.

No estudo de simulação, os dados foram gerados de uma estrutura de média conhecida, assim tornando possível avaliar se o aprendizado dos algoritmos chegaria a alguma configuração próxima àquela que gerou a média da informação. Com a avaliação da verossimilhança dos parâmetros tem-se que mantendo a configuração geradora dos dados não há um único ponto ótimo e por isso os parâmetros estimados podem ser diferentes e ainda assim obterem a mesma plausibilidade na verossimilhança.

Com os algoritmos tradicionais, fez-se 50 estimações e os algoritmos *sag* e *backprop* não conseguiram gerar todas as estimações. Os algoritmos *rprop+* e *rprop-* necessitaram de menor número de iterações para convergir se comparados ao *slr,* em contrapartida, o erro do slr foi inferior aos demais. A maior dispersão de valores estimados nos algoritmos ocorreu nos interceptos e nos parâmetros que ligam os neurônios da camada oculta a de saída.

Todos os algoritmos estimaram valores diferentes daqueles que geraram os dados e obtiveram uma maior verossimilhança também, o que indica que chegaram a uma boa estimativa. Os valores estimados para os parâmetros pelos algoritmos *rprop+* e *rprop-* tiveram maior relação do que com o *slr*.

Com a Inferência Bayesiana foi possível reafirmar a dificuldade em estimar um único parâmetro para uma estrutura de rede neural. As cadeias, no geral, não obtiveram, na distribuição *a posteriori,* média coincidente com a que gerou os dados, além de uma cadeia muitas vezes ser bimodal, o que indica que encontrou dois valores ótimos.

Com o algoritmo proposto fez-se uma combinação do *ranking SIC* com as otimizações dos parâmetros da camada de entrada pelo SA e os da camada oculta para a saída com a regressão linear, com isso obteve-se erro semelhante aos

algoritmos tradicionais, porém, o ganho está na obtenção de uma estrutura muito mais simples.

Foram feitas também duas aplicações em dois estudos de caso. O primeiro é uma série histórica diária de preços de petróleo, a qual possui 1.262 observações que foram separada em período de treinamento e validação. A segunda são dados mensais de exportação de alimentos no Brasil, em R$/US$, para esta as covariáveis utilizadas foram IPCA de alimentos e bebidas, taxa de câmbio R$/US$, câmbio contratado em exportações, imposto sobre a importação e exportações, cada variável com 274 observações.

No primeiro estudo de caso os algoritmos conseguiram capturar o comportamento da série e, entre os tradicionais, o *rprop-* apresentou menor erro no período de treinamento, porém, na previsão obteve-se menor erro com o algoritmo *slr*. O modelo proposto foi comparado então com o *slr* no período de validação a fim de considerar a comparação mais assertiva e ambos obtiveram o mesmo valor de erro, porém, é importante evidenciar que o modelo *slr* obteve dificuldades de convergência em outras configurações.

Ao avaliar os algoritmos que obtiveram convergência em todas as estruturas, tem-se uma configuração de 36 parâmetros no melhor modelo para ambos algoritmos *rprop+* e *rprop-*, o modelo proposto com a avaliação também de todos os neurônios obteve uma estrutura final de 10 parâmetros, o que constitui em uma estrutura muito mais simples. Para este estudo de caso as previsões foram satisfatórias.

Com a inferência Bayesiana as cadeias não obtiveram mesma convergência em todas as configurações de rede utilizadas. O intervalo de credibilidade, apesar de acompanhar a série em diversos pontos não contemplou os valores reais.

O comportamento das variáveis para o segundo estudo de caso não foi absorvido por nenhum dos modelos. Os modelos tradicionais obtiveram bastante erros assim como o modelo proposto e o intervalo de credibilidade da inferência Bayesiana, o que indica a possibilidade de as covariáveis escolhidas para a explicação não serem apropriadas ou então o fato de os dados serem mensais acarretaram perda de informação para previsão.

Com as aplicações, há evidências de que o método proposto possui bons resultados em séries históricas e conseguiu captar informações com uma estrutura muito mais enxuta de rede neural. A aplicação em dados não históricos deve ser

melhor avaliada após outras aplicações com dados que se possa prever com maior assertividade.

5.1 RECOMENDAÇÕES PARA TRABALHOS FUTUROS

Para trabalhos posteriores o método proposto pode ser aplicado em outros conjuntos de dados históricos com diferentes funções de ativação. Também podem ser feitas aplicações com outras arquiteturas de rede neural, além de incluir mais covariáveis.

REFERÊNCIAS

AGGARWAL, CHARU C. **Neural Networks and Deep Learning A Textbook.** Springer, 2018.

ANASTASIADIS, A. D.; MAGOULASA, G. D.; VRAHATISB, M. N. **New globally convergent training scheme based on the resilient propagation algorithm.** Neurocomputing, 2005.

ARAGON, C.R.; JOHNSON D.S.; MCGEOCH L.A.; SCHEVON C. **Optimization by Simulated Annealing: an Experimental Evaluation**, Workshop on Statistical Physics in Engineering and Biology, 1984.

ASADZADEH, F.; MALEKI-KAKELAR, M.; SHABANI, F. **Predicting cationic exchange capacity in calcareous soils of East-Azerbaijan province, northwest Iran.** Communications in Soil Science and Plant Analysis, 50(9), pp.1106-1116, 2019.

ASHRAFIAN, A.; AMIRI, M.J.T.; REZAIE-BALF, M.; OZBAKKALOGLU, T.; LOTFI-OMRAN, O. **Prediction of compressive strength and ultrasonic pulse velocity of fiber reinforced concrete incorporating nano silica using heuristic regression methods**. Construction and Building Materials, 190, pp.479-494, 2018.

BELCIUG, S.; SANDITA, A. **Business Intelligence: statistics in predicting stock market**. Annals of the University of Craiova-Mathematics and Computer Science Series, 44, pp.292-298, 2017.

BELISLE, C. J. P. **Convergence theorems for a class of simulated annealing algorithms on R^d**. Journal of Applied Probability, 29, 885—895, 1992.

BONAT, W. H.; JR RIBEIRO, P. J.; KRAINSK, E. T.; ZEVIANI, W. M. **Métodos Computacionais para Inferência Estatística SINAPE**: Simpósio Nacional de Probabilidade e Estatística, 2012.

BONAT, W. H.; OLIVEIRO, J.; GRANDE-VEGA, M.; FARFÁN, M. A.; FA, J. **Modelling the covariance structure in marginal multivariate count models: Hunting in Bioko Island**. Journal of Agricultural, Biological, and Environmental Statistics, 2017.

BRAGA, A. D. P.; CARVALHO, A. C. P. D. L. F.; LUDEMIR, T. B. **Redes Neurais Artificiais: Teoria e Aplicações** (2 ed.). Rio de Janeiro: LTC, 2000.

BUI, D.T.; NGO, P.T.T.; PHAM, T.D.; JAAFARI, A.; MINH, N.Q.; HOA, P.V.; SAMUI, P., 2019. **A novel hybrid approach based on a swarm intelligence optimized extreme learning machine for flash flood susceptibility mapping.** Catena, pp.184-196, 2019
.

BURNHAM, K. P.; ANDERSON, D. R. **Model Selection and Inference: A practical Information-Theoretic Approach**. New York: Springer-Verlag, 1999.

CABANEROS, S.M.S.; CALAUTIT, J.K.S.; HUGHES, B.R. **Hybrid artificial neural network models for effective prediction and mitigation of urban roadside NO2 pollution**. Energy Procedia, 142, pp.3524-3530, 2017.

CARLIN, B. P.; LOUIS, T. A. **Bayesian methods for data analysis**. Chapman & Hall, 2009.

CASELLA, G.; BERGER, R. L. **Inferência Estatística**. Cengage, 2010.

CHEN, M.L.; DODDI, A.; ROYER, J.; FRESCHI, L.; SCHITO, M.; EZEWUDO, M.; KOHANE, I.S.; BEAM, A.; FARHAT, M. **Beyond multidrug resistance: Leveraging rare variants with machine and statistical learning models in Mycobacterium tuberculosis resistance prediction**. EBioMedicine, v. 43, pp.356-369, 2019.

COSTACHE, R.; AND BUI, D.T. **Spatial prediction of flood potential using new ensembles of bivariate statistics and artificial intelligence: A case study at the Putna river catchment of Romania**. Science of The Total Environment, pp.1098-1118, 2019.

DA SILVA BISPO, V.D.; SCHEID, C.M.; CALÇADA, L.A.; DA CRUZ MELEIRO, L.A. **Development of an ANN-based soft-sensor to estimate the apparent viscosity of water-based drilling fluids.** Journal of Petroleum Science and Engineering, 150, pp.69-73, 2017.

FAUSETT, L. **Fundamentals of Neural Network**. Prentice Hall, Hoboken, 1994.

FRANZIN, A.; STÜTZLE, T. **Revisiting simulated annealing: A component-based analysis.** Computers & Operations Research. V. 104. p.p 191-206, 2019.

GELMAN, A; CARLIN, JB; STERN, HS; DUNSON, DB; VEHTARI, A; RUBIN, DB. **Bayesian data analysis**. Chapman & Hall, 2014.

GIL, A. C. **Como elaborar projetos de pesquisa**. 4a. ed. São Paulo: Atlas, 2002.

GIL, A. C. **Métodos e técnicas de pesquisa social**. 6a. ed. São Paulo: Atlas, 2008.

GÜNTHER, F.; FRITSCH, S. **neuralnet: Training of Neural Networks**. The R Journal, 2010.

HAYKIN, S. **Neural Networks and Learning Machines**. (3 ed.). Prentice Hall, 2009

HAYKIN, S. **Redes Neurais: princípios e práticas**. Tradução de Paulo Martins Engel (2 ed.). Porto Alegre: Bookman, 2001.

GOMES, L. E. S**. Package 'ipeadatar'**. Disponível em: <https://cran.rproject.org/web /packages/ipeadatar/ipeadatar.pdf> Acesso em: 23 jul. 2019.

GÜNAY, M.E. **Forecasting annual gross electricity demand by artificial neural n etworks using predicted values of socio-economic indicators and climatic cond itions: Case of Turkey**. Energy Policy, 90, pp.92-101, 2016.

HEIDARI, E.; SOBATI, M.A.; MOVAHEDIRAD, S. **Accurate prediction of nanofluid viscosity using a multilayer perceptron artificial neural network (MLP-ANN).** Ch emometrics and intelligent laboratory systems, pp.73-85, 2016.

HUSSAIN, S.; ALALILI, A. **A hybrid solar radiation modeling approach using wav elet multiresolution analysis and artificial neural networks**. Applied Energy, v. 20 8, pp.540-550, 2017.

KABESHOVA, A; LAUNAY, C.P.; GROMOV, V.A.; FANTINO, B.; LEVINOFF, E.J.; ALLALI, G.; BEAUCHET. **O. Falling in the elderly: Do statistical models matter for performance criteria of fall prediction? Results from two large population-based studies**. European journal of internal medicine, p. 48-56, 2016.

KE, S.W.; LIN, W.C.; TSAI, C.F.; HU, Y.H. **Soft estimation by hierarchical classification and regression**. Neurocomputing, 234, pp.27-37, 2017.

KIRKPATRICK, S.; GELATT, Jr., C.D.; VECCHI, M.P. **Optimization by Simulated Annealing**. Science, v. 220, p. 671-680, 1983.

KHALIFEH, A.; VAFERI, B. **Intelligent assessment of effect of aggregation on thermal conductivity of nanofluids—Comparison by experimental data and empirical correlations.** Thermochimica Acta, 681, p.178377, 2019.

KHASHEI, M., & BIJARI, M. **An artificial neural network (p , d , q) model for timeseries forecasting**. Expert Systems With Applications, p. 479–489, 2010.

LOUTFI, H.; BERNATCHOU, A.; TADILI, R. **Generation of horizontal hourly global solar radiation from exogenous variables using an artificial neural network in Fes (Morocco).** International Journal of Renewable Energy Research, v.7, p.11., 2017.

METROPOLIS, N; ARIANNA, W. R.; MARSHALL, N. R.; AUGUSTA, H. T.; EDWARD, T.. **Equation of state calculations by fast computing machines**. The journal of chemical physics 21.6, 1087-1092, 1953.

MIRBAGHERI, S.A.; MOHAMMADI, M. **Prediction of environmental effects in received signal strength in FM/TV station based on meteorological parameters using artificial neural network and data mining.** Journal of environmental management, 250, p.109454, 2019.

MÓDOLO, M. **Classificação automática de supernovas usando redes neurais artificiais. Instituto Nacional de Pesquisas Espaciais**. São José dos Campos, 2016.

MOLLALO, A.; MAO, L.; RASHIDI, P.; GLASS, G.E. **A GIS-based artificial neural network model for spatial distribution of tuberculosis across the continental**

United States. International journal of environmental research and public health, p.157, 2019.

MONTGOMERY, C. D.; KULAHCI, M; JENNINGS, C.L., **Introduction to Time Series Analysis and Forecasting**, 2008.

NAVAS, R.K.B.; PRAKASH, S.; SASIPRABA, T. **Artificial Neural Network based computing model for wind speed prediction: A case study of Coimbatore, Tamil Nadu, India.** Physica A: Statistical Mechanics and its Applications, p.123383, 2020.

NUAMAH, I. F., QU, Y.; AMINI, S. B. **A SAS macro for stepwise correlated binary regression**. Computer Methods and Programs in Biomedicine 49, 199–210, 1996.

OBRZUT, B.; KUSY, M.; SEMCZUK, A.; OBRZUT, M.; KLUSKA, J. **Prediction of 5–year overall survival in cervical cancer patients treated with radical hysterectomy using computational intelligence methods**. BMC cancer, p.840, 2017.

PARVEEN, N.; ZAIDI, S.; DANISH, M. **Development and analyses of data-driven models for predicting the bed depth profile of solids flowing in a rotary kiln.** Advanced Powder Technology, 2019.

PAULINO, CD; TURKMAN, MAA; MURTEIRA, B; SILVA, GL. **Estatística bayesiana**. Fundação Calouste Gulbenkian, 2018.

PINO-MEJÍAS, R.; PÉREZ-FARGALLO, A.; RUBIO-BELLIDO, C.; PULIDO-ARCAS, J.A. **Artificial neural networks and linear regression prediction models for social housing allocation: Fuel Poverty Potential Risk Index.** Energy, v.164, pp.627-641, 2018.

PLUMMER, M. **JAGS Version 4.3.0 user manual**, 2017.

QUILES, M. G. **Sistema de Visão Baseado em Redes Neurais para o Controle de Robôs Móveis**. 2004. Dissertação (Mestrado em CIência da Computação e Matemática Computacional) - Universidade de São Paulo. São Paulo. 2004.

R CORE TEAM (2020). **R: A language and environment for statistical computin**. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.o rg/.

RIBEIRO, M.H.D.M.;DOS SANTOS COELHO, L. **Ensemble approach based on bagging, boosting and stacking for short-term prediction in agribusiness time series**. Applied Soft Computing, v. 86, p.105837, 2020.

RIEDMILLER, M.; BRAUN, H. **A direct method for faster backpropagation learnin g: the rprop algorithm**. Proceedings of the IEEE International Conference on Neural Networks (ICNN), 1:586–591, 1993.

RUMELHART, D.E.; MCCLELLEND, J.L. **Parallel Distributed Processing: Explorations in the Microstructure of Cognition**, MIT Press, Cambridge, 1986, pp. 318–362.

SEGURA-BEDMAR, I.; COLÓN-RUÍZ, C.; TEJEDOR-ALONSO, M.Á.; MORO-MORO, M. **Predicting of anaphylaxis in big data EMR by exploring machine learning approaches**. Journal of biomedical informatics, p.50-59, 2018.

ŞENYURT, M.; ERCANLI, I. **A comparison of artificial neural network models and regression models to predict tree volumes for crimean black pine trees in Cankiri forests**. Šumarski list,p.413-423, 2019.

SILVA, E. L.; MENEZES, E. M. **Metodologia da Pesquisa e Elaboração de Dissertação**. 4ª ed., UFSC, Florianópolis, SC, 138p, 2005.

STOKLOSA, J.; GIBB, H.; WARTON, D. I. **Fast forward selection for generalized estimating equations with a large number of predictor variables**, Biometrics 70, 110–120, 2014.

THACH, N.N.; NGO, D.B.T.; XUAN-CANH, P.; HONG-THI, N.; THI, B.H.; NHAT-DUC, H.; DIEU, T.B. **Spatial pattern assessment of tropical forest fire danger at Thuan Chau area (Vietnam) using GIS-based advanced machine learning algorithms: A comparative study.** Ecological Informatics, 46, pp.74-85, 2018.

YAKUBU, A.; OLUREMI, O.I.A.; EKPO, E.I. **Predicting heat stress index in Sasso hens using automatic linear modeling and artificial neural network.** International journal of biometeorology, pp.1181-1186, 2018.

YU, L.; CHEN, J.; DING, G.; TU, Y.; YANG, J.; SUN, J. **Spectrum prediction based on Taguchi method in deep learning with long short-term memory**. IEEE Access, pp.45923-45933, 2018.

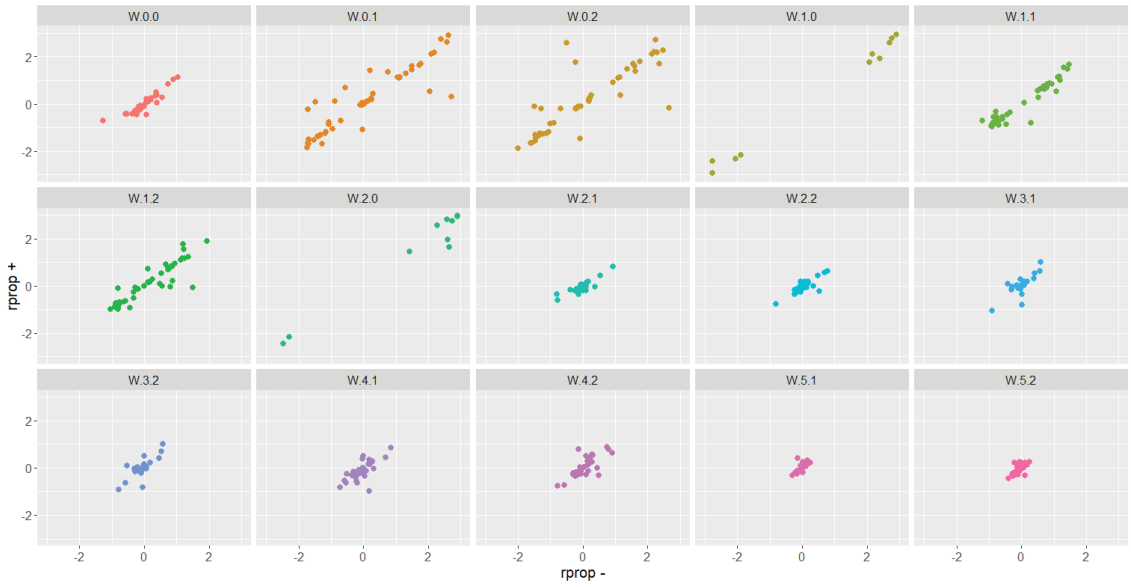
YU, S.; WU, S.; WANG, L.; JIANG, F.; XIE, Y.; LI, L. **A shallow convolutional neural network for blind image sharpness assessment**. PloS one, 2017.

WEI, C.C. **Comparing single-and two-segment statistical models with a conceptual rainfall-runoff model for river streamflow prediction during typhoons.** Environmental Modelling & Software, 85, pp.112-128, 2016.

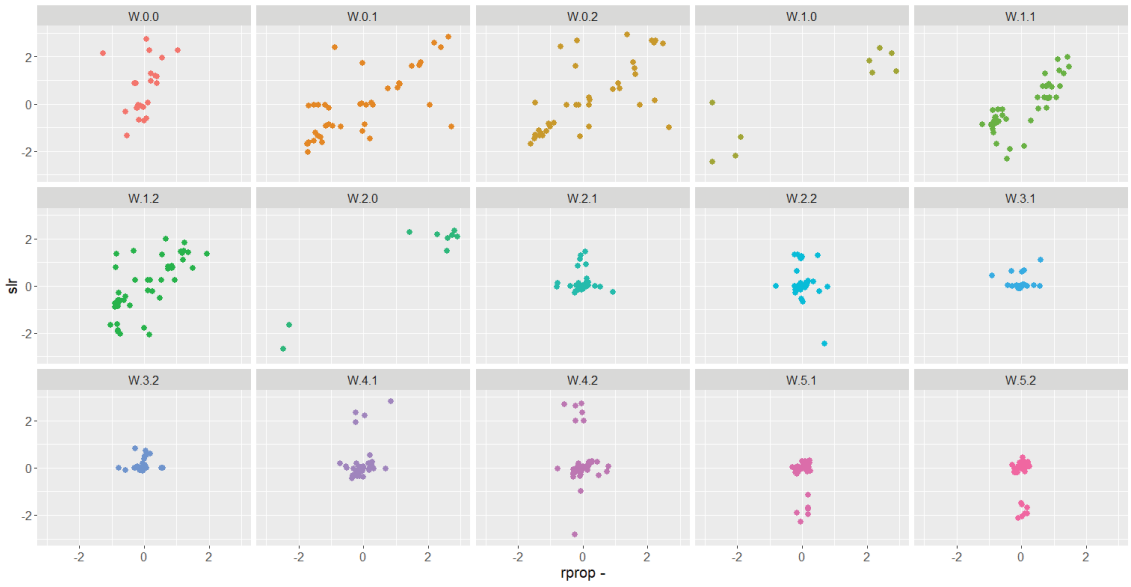
XU, Z.; ZHANG, Q.; LI, W.; LI, M.; YIP, P.S.F. **Individualized prediction of depress ive disorder in the elderly: A multitask deep learning approach.** International Jou rnal of Medical Informatics, 132, p.103973, 2019.

ZANETTI, S. S.; SOUSA, E. F.; CARVALHO, D. F. DE & BERNARDO, S. **Reference evapotranspiration estimate in Rio de Janeiro State using artificial neural networks**, p.174–180, 2008.

APÊNDICE 1 – DIAGRAMA DE DISPERSÃO RPROP + E RPROP-



APÊNDICE 2 – DIAGRAMA DE DISPERSÃO SLR E RPROP-



APÊNDICE 3 – DIAGRAMA DE DISPERSÃO *SLR* E *RPROP* +